计算机科学丛书

原书第2版

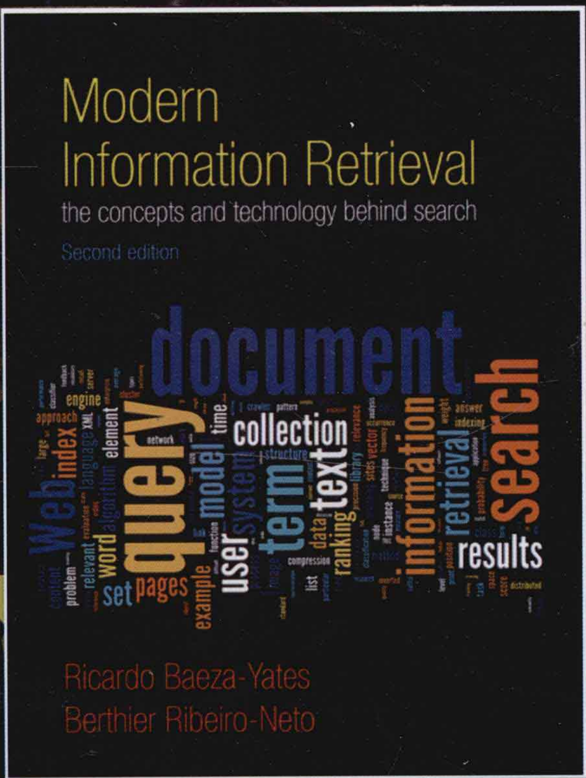
现代信息检索

Ricardo Baeza-Yates　Berthier Ribeiro-Neto　著

黄萱菁　张奇　邱锡鹏　译

Modern Information Retrieval

The Concepts and Technology behind Search　Second Edition



机械工业出版社
China Machine Press

VISIT…

LANZAROTE Caliente.com

现代信息检索（原书第2版）

Modern Information Retrieval The Concepts and Technology behind Search Second Edition

本书详细介绍了信息检索的所有主要概念和技术，以及有关信息检索方面的所有新变化，使读者既可以对现代信息检索有一个全面的了解，又可以获取现代信息检索所有关键主题的详细知识。本书的主要内容由信息检索领域的代表人物Baeza-Yates和Ribeiro-Neto撰写，对于那些希望深入研究关键领域的读者，书中还提供了由其他主要研究人员撰写的关于特殊主题的发展现状。

与上一版相比，本版在内容和结构上都有大量调整、更新和充实，其中新增内容在60%~70%左右。具体更新情况如下：

- 新增了文本分类、Web爬取、结构化文本检索和企业搜索等章节，以及关于开源搜索的一个附录。
- 全面改写了用户界面、多媒体检索和数字图书馆等内容。
- 拓展了一些章节，介绍了信息检索方面的新的重要进展，如语言模型、新的评价方法、查询的特点、基于集群的信息检索和分布式信息检索等。

作者简介



Ricardo Baeza-Yates 于加拿大滑铁卢大学获得计算机科学博士学位，现为雅虎欧洲和拉丁美洲研究院副总裁，主管雅虎在巴塞罗纳（西班牙）和圣地亚哥（智利）的研究中心，并监管海法研究中心。他曾担任智利计算机科学学会主席、智利大学计算机科学系Web研究中心主任、ICREA教授，并且他还在巴塞罗纳法布拉大学创立了信息与通信技术系Web研究组。现在他仍是智利大学和法布拉大学的兼职教授。他的主要研究方向为算法与数据结构、信息检索、用户界面以及可视化在数据库中的应用等。



Berthier Ribeiro-Neto 于加利福尼亚大学洛杉矶分校获得计算机科学博士学位，现任巴西Minas Gerais联合大学计算机科学系副教授，同时也是ACM、ASIS及IEEE会员。他的主要研究方向是信息检索系统、数字图书馆、Web界面及视频点播。



书号：978-7-111-33174-2
定价：78.00元

客服热线：(010) 88378991, 88361066
购书热线：(010) 68326294, 88379649, 68995259
投稿热线：(010) 88379604
读者信箱：hzjsj@hzbook.com

华章网站 http://www.hzbook.com

网上购书：www.china-pub.com

封面设计·锡彬

上架指导：计算机/信息检索

ISBN 978-7-111-38599-8

9 787111 385998

定价：118.00元

计算机科学丛书

原书第2版

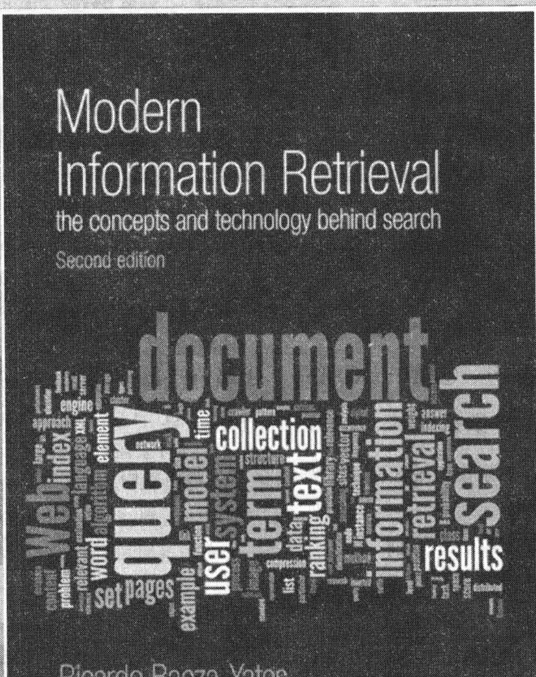
现代信息检索

Ricardo Baeza-Yates　Berthier Ribeiro-Neto　著

黄萱菁　张奇　邱锡鹏　译

Modern Information Retrieval

The Concepts and Technology behind Search Second Edition



机械工业出版社
China Machine Press

本书论述信息检索的概念和技术、这些技术在搜索引擎中的应用，及其对相关领域知识的影响等，主要内容包括：用户界面设计；经典的信息检索模型、结果质量评估和用户相关反馈；文档和查询概念及其相关技术；文档集索引和搜索技术；Web文档的爬取、检索和排序；结构化文本检索、多媒体检索和企业搜索；图书馆系统和数字图书馆等。

本书内容广泛、细节丰富、深入浅出，可以作为高等院校信息管理与信息系统、计算机科学与技术、图书馆学、情报学、档案学等专业本科生和研究生的教材或参考书，对从事信息检索及系统分析、设计的实际工作者也有较高的参考价值。

Ricardo Baeza-Yates, Berthier Ribeiro-Neto: Modern Information Retrieval: The Concepts and Technology behind Search, Second Edition (9780321416919).

Copyright © 2011 by Pearson Education Limited.

This translation of Modern Information Retrieval: The Concepts and Technology behind Search, Second Edition (9780321416919) is published by arrangement with Pearson Education Limited.

All rights reserved.

本书中文简体字版由英国 Pearson Education 培生教育出版集团授权出版。

封底无防伪标均为盗版
版权所有，侵权必究
本书法律顾问　北京市展达律师事务所

本书版权登记号：图字：01-2010-6144

图书在版编目（CIP）数据

现代信息检索（原书第2版）/（智）贝泽-耶茨（Baeza-Yates, R.）等著；黄萱菁，张奇，邱锡鹏译. —北京：机械工业出版社，2012.8
（计算机科学丛书）
书名原文：Modern Information Retrieval: The Concepts and Technology behind Search, Second Edition

ISBN 978-7-111-38599-8

Ⅰ.现…　Ⅱ.①贝…　②黄…　③张…　④邱…　Ⅲ.情报检索　Ⅳ.G252.7

中国版本图书馆 CIP 数据核字（2012）第114931号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：盛思源
藁城市京瑞印刷有限公司印刷
2012年10月第1版第1次印刷
185mm×260mm · 43.25 印张
标准书号：ISBN 978-7-111-38599-8
定价：118.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991；88361066
购书热线：（010）68326294；88379649；68995259
投稿热线：（010）88379604
读者信箱：hzjsj@hzbook.com

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章公司较早意识到“出版要为教育服务”。自 1998 年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与 Pearson, McGraw-Hill, Elsevier, MIT, John Wiley & Sons, Cengage 等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出 Andrew S. Tanenbaum, Bjarne Stroustrup, Brain W. Kernighan, Dennis Ritchie, Jim Gray, Afred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, Abraham Silberschatz, William Stallings, Donald E. Knuth, John L. Hennessy, Larry L. Peterson 等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

华章网站：www.hzbook.com
电子邮件：hzjsj@hzbook.com
联系电话：(010) 88379604
联系地址：北京市西城区百万庄南街 1 号
邮政编码：100037

HZ BOOKS
华章教育
华章科技图书出版中心

译者序

十多年前，我刚刚开始接触信息检索，读了几本经典教材，也看了不少论文，但因为缺乏有关信息检索系统实现的文献，上手很慢。同学从国外回来，带来了 Ricardo Baeza-Yates 撰写的《Information Retrieval: Data Structures and Algorithms》，该书系统地介绍了信息检索领域的重要数据结构和算法，可操作性极强，我简直是如获至宝，也因而记下了 Ricardo 的大名。

几年后，Ricardo 和 Berthier 合著了本书的第 1 版，拜读之后，惊叹于作者不仅具备娴熟的实践技巧，深厚的理论功底，而且还有很强的大局观、洞察力和驾驭素材的能力。该书毫无疑问地成为复旦大学研究生课程"信息检索"的首选教材。

去年春天，好友秦兵教授告诉我，机械工业出版社引进了这本书的第 2 版，打算翻译成中文版，如果我有兴趣，她可以向出版社推荐。虽然此前从未翻译过任何书籍，自己的工作负担也已很重，但出于对本书及作者的推崇，我毫不犹豫地接下了这份任务。

收到出版社寄来的样书后，我发现第 2 版与第 1 版相比可谓截然不同。应该说本书的第 1 版已经足够优秀，被世界上数以百计的大学和学校采纳为教科书，但两位作者仍然大刀阔斧地对许多章节进行了彻头彻尾的修改，并增加了许多新的章节，第 2 版的 60%～70%由新的素材组成即是印证。

第 2 版的巨大变化来自于以下原因：第一，随着互联网的普及，搜索引擎进入人们的日常生活中，成为获取信息的重要入口，用户需求带动了搜索引擎产业的飞速发展，谷歌、雅虎、必应和百度等企业成长为极有影响力的互联网公司，作者因而在本书中加入了许多和搜索引擎有关的章节，如搜索引擎界面、并行和分布式检索、Web 爬取等；第二，产业界的繁荣吸引了大量的研究人员和从业者，而搜索引擎的普及带来了海量的真实用户数据，这些都极大地促进了信息检索研究水平的提高，本书为此增加了语言模型、排序学习等新的研究内容；第三，撰写第 1 版的时候，作者还是大学教师，在撰写第 2 版之际，他们开创了自己的搜索事业，之后进入了主流搜索引擎公司工作，丰富的经历带来更开阔的视野，对搜索引擎也有了更深入的了解。第 2 版不仅反映了信息检索产业界和学术界的变化，也体现了他们在研究、开发和实现信息检索技术，并将其应用于互联网过程中的心得体会。

本书主要由黄萱菁、张奇和邱锡鹏三人执笔翻译。周雅倩、王秉卿、计峰、丁卓冶、吴龑、周金龙和刘昭等同事和研究生帮助做了许多资料整理、录入、校对等辅助工作，李伟和路红两位同事帮助我们了解了多媒体检索所特有的许多概念，王春华、盛思源两位编辑帮助发现了译稿中的许多不足之处，本书两位原作者帮助澄清了许多问题，复旦大学计算机学院为本书的翻译提供了有力支持，在此一并致谢。

翻译一本书，比我想象的要困难很多。好的译者，不仅要对领域知识有充分的了解和掌握，也需要流畅精彩的文笔。然而，"知易行难"，本书的几位译者都是理工科出身，虽然都是具有一定经历的信息检索研究人员，但第一次从事翻译工作，水平有限，错漏之处在所难免，敬请各位读者谅解并批评指正。

黄萱菁
2012 年春于浦东张江

第 2 版前言

自从本书第 1 版出版以来，信息检索（Information Retrieval，IR）领域发生了许多变化，其中许多和 Web 有关。首先，Web 上的海量信息已将搜索引擎转化为寻找和发现用户感兴趣信息的关键工具。其次，由于搜索引擎的本质核心是信息检索系统，这就有力地证明了信息检索技术可以应用于具有巨大查询流量的海量文档集。

紧随这一演变趋势，在本书第 1 版出现以后的短短几个月内，我们在巴西和智利就开始了搜索引擎的研究。后来，我们进入谷歌和雅虎这两个主要的搜索引擎公司工作，对搜索引擎的一切行为有了更深入的了解。因此，本书第 2 版不仅反映了信息检索领域的变化，也反映了我们自己正在研究、开发和实现的信息检索技术，以及将其应用于 Web 的经验。

本书第 1 版并不是按照标准方式书写的，对于我们觉得没有足够专业知识的领域，我们邀请专家撰写相关章节。所以，从某种意义上说，我们先于 Web 2.0 的发展趋势进行了团队协作。我们的宗旨是精心协调和监督所有的写作内容，使本书成为有机的整体。在某种程度上，我们的努力颇有成效。事实上，第 1 版卖得非常好，成为了信息检索领域的畅销书，并已重印多次。该书已被数以百计的大学和学校采纳。它首先被翻译成韩文，其次是中文，还有一个特别低价的版本已在印度出版。因此，第 1 版出版后仅仅一两年，我们就开始谈论第 2 版。这个想法一直到 2004 年我们向出版商提交建议书并获得批准后才得以实现。最终在 2005 年 11 月，也就是四年多前，我们开始第 2 版的工作。今天，我们终于完成了！

在第 2 版中，我们遵循着和第 1 版相同的方法，因为它明显行之有效。尽管如此，我们仍然是更多章节的作者或合著者，而且我们采取了更强有力的手段对其他章节的内容进行设计。我们不得不完全修改许多章节，并增加了许多新的章节。因此，第 2 版的 60%～70% 是由新素材组成的，和第 1 版的不同之处主要在以下几个方面：

- 完全重组第 1 章内容。
- 增加文本分类、Web 爬取、结构化文本检索和企业搜索等新章节，以及一个关于开源搜索引擎的新附录。
- 完全重写用户界面、多媒体检索和数字图书馆等章节。
- 扩充章节内容，以包括重要的新进展，例如语言模型、新的评价准则、查询特性、基于集群的信息检索和分布式信息检索、排序学习、搜索引擎界面和个性化等。
- 改进本书网站，其中包括本书所有章节的全套幻灯片和推荐的练习列表，使之成为信息检索的参考教学资源。

最后的成果是，和第 1 版相比，第 2 版几乎有两倍的篇幅，并包含两倍以上的参考文献。总之，如果你喜欢本书第 1 版，我们希望你会更喜欢这个第 2 版。万一你不喜欢第 1 版，我们希望这一次你会改变主意。

Ricardo Baeza-Yates 于西班牙巴塞罗那
Berthier Ribeiro-Neto 于巴西贝洛奥里藏特
2010 年 12 月

第1版前言

随着Web的发展，以及时尚而廉价的图形用户界面和海量存储设备的问世，信息检索在过去几年中发生了巨大的变化。传统的信息检索教科书已相当过时，为此，最近已经出版了一些新的信息检索书籍。不过，我们相信，仍然非常需要这样一本书，它能够从计算机科学的视角，而不是从用户为中心的视角，以严密和完整的方式来介绍这个领域。本书致力于部分地填补这一鸿沟，它既可以作为信息检索的入门教材，也可以用于该方向的研究生课程。

本书是由相互补充和平衡的两部分组成。核心部分包括由本书设计者撰写或合著的9章。第二部分和第一部分紧密相连，共分为6章。这部分由相关领域的领先研究人员撰写，介绍最新的研究进展。所有章节采用相同的符号和术语。因此，尽管事实上邀请了多位撰稿人，但这本书并不是由不同作者撰写的章节汇编成的合著，而是一本教科书。此外，与合著相比，本书的主要作者精心设计了全书的内容和结构，以便展示现代信息检索中所有重要方面的内在联系。

从信息检索模型到文本索引，从信息检索可视化工具和界面到Web，从多媒体信息检索到数字图书馆，本书都广泛涵盖，而且细节丰富。考虑到信息检索对现代社会显而易见的相关性和重要性，我们希望本书对世界各地的信息科学、计算机科学与图书馆学等学科研究的进一步传播起到促进作用。

Ricardo Baeza-Yates 于智利圣地亚哥
Berthier Ribeiro-Neto 于巴西贝洛奥里藏特
1998年10月

第 2 版致谢

我们对在过去几年间向我们提供了有用和有益的意见、评论和建议的人们致以衷心的感谢。本书内容和素材组织的改进，很大程度上归功于他们。如果没有他们的帮助，第 2 版的质量将大大下降。仍然存在的任何错误——希望只有少量，完全是我们的责任。

第一，我们对所有撰稿人所体现出的奉献精神和浓厚兴趣表示感谢，他们是 Eric Brown、Carlos Castillo、Marcos Gonçalves、David Hawking、Marti Hearst、Mounia Lalmas、Yoelle Maarek、Christian Middleton、Gonzalo Navarro、Dulce Ponceleón、Edie Rasmussen、Malcolm Slaney 和 Nivio Ziviani。他们所体现的专业知识是我们所欠缺的。

第二，我们感谢对第 2 版的新内容提供直接或者间接贡献或影响的人们，他们是 Omar Alonso（他指出我们偏离了众包的重要趋势）、Paolo Boldi（Web 图压缩）、Pavel Calado（文本分类）、Marco Cristo（他对于文本分类章节的意见导致了对素材的整体重组）、Christos Faloutsos（多维索引）、Winston Hsu（多媒体）、Flavio Junqueira（分布式检索）、Edleno Moura（检索评价）、Vanessa Murdock（查询困难性）、Martin Porter（词干提取算法）、Mark Sanderson（他的尖锐意见导致检索评价章节的重大改进）、Fabrizio Silvestri（URL 排序）和 Gleb Skobeltsyn（对等网络信息检索）。另外，我们还感谢巴西米纳斯吉拉斯州联邦大学 Marcos Gonçalves 的多位研究生的贡献，他们评阅了文本分类章节并书写了大量意见。

第三，我们需要感谢所有提供第 1 版勘误信息、提出改进建议和对第 2 版草稿提出修改意见的人们。对于勘误表，我们只提及发现错误的第一人，否则名单将太长。他们是：Omar Alonso、Jose Hilario Canos、Berkant Barla Cambazoglu、Ernie Davis、Anne Diekema、Bill Dimm、Joaquim Gabarro、Jamie Geddes、Eduardo Graells、Kyoung-Soo Han、Claudia Hauff、Shoujie He、Ben Houston、Puay-Leng Lee、Songwook Lee、Shian-Hua Lin、Mildrid Ljosland、Chang-Tien Lu、Mari Carmen Marcos、Peter Mika、Vanessa Murdock、Joanna Plattner、Luz Rello、Hee-Cheol Seo、Ben Shneiderman、Helge Grenager Solheim、Ellen Spertus、Markus Stocker、Kazunari Sugiyama、Satoru Takabayashi、Juha Takkinen、Luong Minh Thang、Yannis Tzitzikas、Fredrik Wallenberg、Theo van der Weide、John Westbrook、Judith Winter、Sui Xi、Peng Yong、Hugo Zaragoza 和 Yonghui Zhang。上述名单可能不全。

第四，我们特别感谢 David Fernandes，本书网站上有他制作的教学幻灯片。他也耐心指出了许多小错误和不一致的地方。我们也需要提及我们的雇主雅虎和谷歌，他们为我们完成撰写本书的艰巨任务提供了隐性支持。

第五，我们感谢 Pearson Education 公司的编辑。他们是 Kate Brewin、Simon Plumtree、Owen Knight 和 Rufus Curnow。在最重要的出版过程中，他们给予了支持。Anita Atkinson 和 Jenny Oates 分别是本书的文字编辑和校对，我们感谢她们的帮助。

最后也是最重要的，感谢 Helena、Rosa 和我们的孩子，他们再次忍受了我们一连串的国际旅行、周末加班和不规律的工作时间。在过去的 4 年里，他们总是在问：你们什么时候完成这本书？

第1版致谢

我们对在过去几个月的写作过程中向我们提供了有用和有益帮助的各位人士致以衷心的感谢。如果没有他们的关心，本书很可能无法完成。

第一，我们对所有撰稿人所体现出的奉献精神和浓厚兴趣表示感谢。他们是 Elisa Bertino、Eric Brown、Barbara Catania、Christos Faloutsos、Elena Ferrari、Ed Fox、Marti Hearst、Gonzalo Navarro、Edie Rasmussen、Ohm Sornil 和 Nivio Ziviani。他们所体现的专业知识是我们所欠缺的。我们也感谢他们在编辑和交叉审阅过程中给予的耐心，这是一种相当难以平衡的工作。

第二，我们要感谢对出版本书感兴趣的所有人士，特别是 Scott Delman 和 Doug Sery。

第三，对于 Addison Wesley Longman 出版社对我们的兴趣和给予的鼓励，以及在整个过程中所做的优秀工作，我们在此深表感谢。他们的代表是 Keith Mansfield、Karen Sutherland、Bridget Allen、David Harrison、Sheila Chatten、Helen Hodge 和 Lisa Talbot。他们联系的评阅人阅读了本书的早期（也是非常原始的）方案，并提供了很好的反馈意见，显示了深刻的洞察力。鉴于一位匿名评阅人的客观评论，“并行和分布式检索”章节从不很合适的“信息检索应用”部分移到了“文本信息检索”部分。鉴于检索评价的重要性，另一位热心的评阅人强烈建议我们将它单列为一章。

第四，我们要感谢和我们讨论过本书撰写计划的所有人士。Doug Oard 很早就评阅了本书的草案。Gary Marchionini 是本书的早期支持者，并在我们写书的过程中保持联系。Bruce Croft 从一开始就鼓励我们。Alberto Mendelzon 提供了 Web 搜索章节的初始方案和参考文献列表。Ed Fox 在百忙之中对第1章“引言”提出了富有洞察力的评阅意见，使我们极大地改进了这一章。他也认真评阅了信息检索建模的内容。Marti Hearst 很早就对我们的方案深表兴趣，在整个编辑过程中提供了帮助，并且是一个热情的支持者和伙伴。

第五，我们感谢我们所在的机构，智利大学和巴西米纳斯吉拉斯州联邦大学计算机科学系的支持，以及来自国家研究机构——巴西科技发展委员会（CNPq）、智利国家科技研究委员会（CONICYT）和国际合作项目的经费资助，特别是拉美科技发展项目（CYTED）项目“Web 信息管理与检索环境（Environment for Information Managing and Retrieval in the World Wide Web，AMYRI，编号 VII.13）”和巴西科学研究与发展项目资助署（Finep）项目“移动计算机的信息系统（Information Systems for Mobile Computers，SIAM）”。

最重要的是，感谢 Helena、Rosa 和我们的孩子，他们忍受了我们一连串的国际旅行、周末加班和不规律的工作时间。

我们感谢以下复制版权材料的许可：

图

图 2-1 和图 2-12 来自 Yelp!，http://www.yelp.co.uk/，Yelp! Inc.；图 2-3 来自 NextBio.com；图 2-5、图 4-13b、图 11-10c、图 11-11a 和图 11-13 来自 www.google.co.uk 提供的谷歌系统截图；图 2-6 来自 http://biosearch.berkerley.edu，M. A. Hearst 版权所有；图 2-7 来自 Microsoft Corporation 的产品截图重印许可；图 2-13 来自 Findex、FindEx.com，Inc. 及其许可者版权所有©2010；图 2-15 来自 "Graphical query specification and dynamic result previews for a digital library, Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology (UIST'98) pp.143-151 (Jones, S. 1998)"，http://doi.acm.org/10.1145/288392.288595，Association for Computing Machinery，Inc. 版权所有©1998，重印经许可；图 2-16 来自 "Research: TileBars"，http://people.ischool.berkeley.edu/~hearst/research/tilebars.html，M. A. Hearst 版权所有；图 2-17a来自 "Search User Interfaces, Cambridge University Press (Hearst, M. A. 2009)" 的图 10-17a，M. A. Hearst 版权所有；图 2-17b 来自 "INSYDER: a content-based visual-information-seeking system for the web, International Journal on Digital Libraries, pp.25-41 (Reiterer, H., Tullius, G. and Mann, T. M. 2005)"，许可来自 Springer Science + Business Media and CCC 及 H. Reiterer 教授；图 2-18 来自 "Using thumbnails to search the Web, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'01), pp.198-205 (Woodruff, A., Faulring, A., Rosenholtz, R., Morrison, J. and Pirolli, P. 2001)"，http://doi.acm.org/10.1145/365024.365098，Association for Computing Machinery，Inc. 版权所有©2001，重印经许可；图 2-20a 来自 "Evaluating a system for interactive exploration of large, hierarchically structured document repositories, Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'04), pp.127-134 (Granitzer, M., Kienreich, W., Sabol, V., Andrews, K. and Klieber, W. 2004)"，IEEE 版权所有© 2004；图 2-20b 来自 "Search result visualisation with xFIND, Proceedings of User Interfaces to Data Intensive Systems (UIDIS 2001), pp.50-58 (Andrews, K., Gutl, C., Moser, J., Sabol, V. and Lackner, W. 2001)"，IEEE 版权所有©2001；图 2-21 来自 http://kylescholz.com/projects/wordnet/，Kyle Scholz；图 2-22 来自 "The Word tree, an interactive visual concordance, IEEE Transactions on Visualization and Computer Graphics, 14 (6), pp.1221-1228 (Wattenberg, M. and Fernanda, B. 2008)"，IEEE 版权所有© 2008；图 2-23 来自婴儿名字流行度图 NameVoyager，http://www.babynamewizard.com；图 2-24 来自 "Avian flu case study with nSPace and GeoTime, Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST'06) pp.27-34 (Proulx, P. et al. 2006)"，IEEE 版权所有©2006；图 5-4 仿自 "Evaluating the accuracy of implicit feedback from clicks and query reformulations in Web search, ACM Transactions on Information Systems, 25 (2) (Joachims, T., Granka, L., Pan, B., Hembrooke, H.,

Radlinski, F. and Gay, G. 2007)", http://doi.acm.org/10.1145/1229179.1229181, Association for Computing Machinery, Inc. 版权所有©2007, 重印经许可；图 7-4 和图 7-5 来自 "The impact of caching on search engines, Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGR'07) (Baeza-Yates, R. et al. 2007)", http://doi.acm.org/10.1145/1277741.1277775, Association for Computing Machinery, Inc. 版权所有©2007, 重印经许可；图 7-6 来自 "Query usage mining in search engines, Web Mining Applications and Techniques (Baeza-Yates, R. (Scime, A. ed.) 2004)", Idea Group, 重印经出版商 IGI Global 许可；图 10-1 改编自 "Load balancing for term-distributed parallel retrieval, Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 348-355 (Moffat, A., Webber, W. and Zobel, J. 2006)", http://doi.acm.org/10.1145/1148170.1148232, Association for Computing Machinery, Inc. 版权所有©2006, 重印经许可；图 10-12 和图 10-13 来自 "Challenges on distributed web retrieval, Proceedings of ICDE 2007, pp. 6-20 (2007)", IEEE 版权所有©2007；图 10-14 来自 "A pipelined architecture for distributed text query evaluation, Information Retrieval, 10 (3), pp. 205-231 (Webber, W., Moffat, A., Zobel, J. and Baeza-Yates, R. 2007)", 许可来自 Springer Science＋Business Media；图 11-1 来自 "Graph structure in the web: experiments and models, Proceedings of the North Conference on World Wide Web, pp. 309-320 (Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopolan, S., Stata, R., Tomkins, A. and Wiener, J. 2000)", Elsevier 版权所有 (2000)；图 11-3a 来自 M. Crovella, 1998；图 11-3b 来自 "Self-similarity in World Wide Web traffic: evidence and possible causes, SIGMETRICS'96: Proceedings of the 1996 ACM SIGMETRICS International Conference on Measurement and Modelling of Computer Systems, 24, pp. 160-169 (Crovella, M. E. and Bestavros, A. 1996)", http://doi.acm.org/10.1109/90.650143, Association for Computing Machinery, Inc. 版权所有©1996, 重印经许可；图 11-4 和图 11-5 来自 "Generic damping functions for propagating importance in linkbased ranking algorithms, Internet Mathematics, 3 (4), pp. 445-478 (Baeza-Yates, R., Boldi, P. and Castillo, C. 2006)", A. K. Peters, Ltd. 版权所有 2006；图 11-7 仿自 "Challenges in building large-scale information retrieval systems: invited talk presentation", http://research.google.com/people/jeff/WSDM09-keynote.pdf, Jeffrey Dean；图 11-8 来自 "Design trade-offs for search engine caching, TWEB, 2 (4) (Baeza-Yates, R. A., Gionis, A., Juncqueira, F., Murdock, V., Plachouras, V. and Silvestri, F. 2008)", http://doi.acm.org/10.1145/1409220.1409223, Association for Computing Machinery, Inc. 版权所有©2008, 重印经许可；图 11-10a 来自 Ask 系统截图, IAC Search & Media, Inc. 保留所有权利©2010。ASK.COM、ASK JEEVES、ASK 商标、ASK JEEVES 商标及其他出现在 Ask.com 和 Ask Jeeves 网站上的商标属于 IAC Search & Media, Inc. 及其授权者；图 11-10b 及图 11-15 来自 Bing 系统截图, 重印经 Microsoft Corporation 许可；图 12-8 来自 "Synchronizing a database to improve freshness, Proceedings of ACM International Conference on Management of Data (SIGMOD), pp. 117-128 (Cho, J. and Garcia-Molina, H. 2000)", http://doi.acm.org/10.1145/342009.335391, Association for Computing Machinery, Inc. 版权所有©2000, 重印经许可；图 13-9 来自 INEX 2006 评估界面, 由 Mounia Lalmas 教授提供；图 14-4 来自

IBM Almaden 研究中心；图 14-6 和图 14-8 来自 IBM Almaden 研究中心 QBIC 系统，Jim Hafner 的许可；图 14-9 来自 "A bipartite graph model for associating images and text, IJ-CAI-2007 Workshop on Multimodal Information Retrieval (Srinivasan, S. H. and Slaney, M. 2007)"；图 14-10 来自 "Image retrieval on large-scale image databases, Proceedings of the 6th ACM International Conference on Image and Video Retrieval (CIVR 07), pp. 17-24 (Horster, E., Lienhart, R. and Slaney, M. 2007)", http://doi.acm.org/10.1145/1282280.1282283, Association for Computing Machinery, Inc. 版权所有©2007，重印经许可；图 14-13 和图 14-14 来自 Kyogu Lee；图 14-16 来自 Carnegie Mellon 大学计算机学院技术报告 "Video skimming for quick browsing based on audio and image characterization, Technical Report CMU-CS-95-186 (Smith, M. A. and Kanade, T. 1995)"；图 14-17 来自 "Video manga: generating semantically meaningful video summaries, MULTIMEDIA'99: Proceedings of the Seventh ACM International Conference on Multimedia (Part 1), pp. 383-392 (Uchihashi, S. et al. 1999)", http://doi.acm.org/10.1145/319463.319654, Association for Computing Machinery, Inc. 版权所有©1999，重印经许可；图 14-18 来自 Sarnoff Corporation 的 Harpreet Sawhney；图 14-19 来自 "Salient stills, ACM Transactions on Multimedia Computing, Communications and Applications, 1 (1), pp. 16-36 (Teodosio, L. and Bender, W. 2005)", http://doi.acm.org/10.1145/1047936.1047940, Association for Computing Machinery, Inc. 版权所有©2005，重印经许可；图 14-20 来自 "PanoramaExcerpts: Extracting and packing panoramas for video browsing, MULTIMEDIA'97: Proceedings of the Fifth ACM International Conference on Multimedia, pp. 427-436 (Taniguchi, Y., Akutsu, A. and Tonomura, Y. 1997)", http://doi.acm.org/10.1145/266180.266396, Association for Computing Machinery, Inc. 版权所有©1997，重印经许可；图 14-21 来自 "Hierarchical brushing in a collection of video data, Proceedings of Hawaii International Conference on System Science (HICSS) (2001)", IEEE 版权所有©2001；图 14-26 来自 "Automatic recognition of audiovisual speech: recent progress and challenges, Proceedings of the IEEE (Potamianos, G., Neti, C., Gravier, G., Garg, A. and Senior, A. W. 2003)", IEEE 版权所有©2003；图 14-28 来自 "Multimedia edges: finding hierarchy in all dimensions, Proceedings of 9th ACM International Conference on Multimedia (Slaney, M., Ponceleon, D. and Kaufman, J. 2001)", http://doi.acm.org/10.1145/500141.500149, Association for Computing Machinery, Inc. 版权所有©2001，重印经许可；图 14-29 来自 "Comparison of automatic shot boundary detection algorithms, SPIE Image and Video Processing VII, 3656, 290-301 (Lienhart, R. 1999)", SPIE；图 15-3 来自 Oxfam Australia；图 15-5 来自 "Evaluation by comparing result sets in context, Proceedings of the 15th ACM International Conference on Information and Knowledge Management pp. 94-101 (Thomas, P. and Hawking, D. 2006)", http://doi.acm.org/10.1145/1183614.1183632, ACM 版权所有©2006；图 16-1 来自 Edie Rasmussen，许可来自 The Network Development and MARC Standards Office；图 16-2 来自 "Find... books or journals", http://www.library.ubc.ca/home/research.html，不列颠哥伦比亚大学网站 (2010)，许可后使用；图 16-4、图 16-5、图 16-6 和图 16-7 来自 DIALOG，Dialog® 的界面及截屏，经 Dialog LLC. 许可后改编，Dialog 产品名是 Dialog LLC. 的注册商标；图 16-4 来自 EBSCO Publishing，Inc. 的许可。

表

表 4-2 改编自“Overview of the sixth text retrieval conference（TREC-6），Proceedings of the Sixth Text REtrieval Conference（TREC-6）（Voorhees，E. and Harman，D. 1997）”；表 7-3 来自“From e- sex to e- commerce：Web search changes，Computer，35（3），pp. 107-109（Spink，A.，Jansen，B. J.，Wolfram，D. and Saracevic，T. 2002）”，IEEE 版权所有© 2002。

文字

原书 159 页的引文来自 http://trec. nist. gov，NIST。

在某些情况下，我们已经无法追溯版权材料的所有者，读者如能提供任何帮助信息，我们将不胜感激。

出版者的话
译者序
第 2 版前言
第 1 版前言
第 2 版致谢
第 1 版致谢
出版商致谢

第 1 章 引言……………………………… 1
1.1 信息检索 ………………………… 1
1.1.1 信息检索的早期发展 ……… 1
1.1.2 图书馆和数字图书馆中的信息检索 ……………………… 2
1.1.3 舞台中央的信息检索 ……… 2
1.2 信息检索问题 …………………… 3
1.2.1 用户的任务 ………………… 3
1.2.2 信息检索与数据检索 ……… 4
1.3 信息检索系统 …………………… 4
1.3.1 信息检索系统的软件架构 … 4
1.3.2 检索和排序过程 …………… 5
1.4 Web ……………………………… 6
1.4.1 Web 简史 …………………… 7
1.4.2 电子出版时代 ……………… 7
1.4.3 Web 如何改变搜索 ………… 8
1.4.4 Web 上的实际问题 ………… 9
1.5 本书的组织结构 ………………… 9
1.5.1 本书的重点 ………………… 9
1.5.2 本书的内容…………………… 10
1.6 本书的教学资源网站…………… 12
1.7 文献讨论………………………… 12
第 2 章 用户搜索界面 ………………… 16
2.1 介绍……………………………… 16
2.2 人们如何搜索…………………… 16
2.2.1 信息查找与探索式搜索…… 16
2.2.2 信息搜寻的经典模型与动态模型……………………… 17
2.2.3 导航与搜索…………………… 18
2.2.4 对搜索过程的观察………… 18
2.3 现今的搜索界面………………… 19
2.3.1 启动搜寻…………………… 19
2.3.2 查询描述…………………… 19
2.3.3 查询描述界面……………… 20
2.3.4 检索结果显示……………… 22
2.3.5 查询重构…………………… 24
2.3.6 组织搜索结果……………… 26
2.4 搜索界面的可视化……………… 32
2.4.1 可视化布尔语法…………… 32
2.4.2 可视化查询结果中的查询项……………………………… 33
2.4.3 可视化词语和文档间的关系…………………………… 36
2.4.4 文本挖掘的可视化………… 38
2.5 搜索界面的设计和评价………… 40
2.6 趋势和研究问题………………… 42
2.7 文献讨论………………………… 42
第 3 章 信息检索建模 ………………… 44
3.1 信息检索模型…………………… 44
3.1.1 建模和排序………………… 44
3.1.2 信息检索模型描述………… 44
3.1.3 信息检索模型的分类体系 … 45
3.2 经典信息检索…………………… 47
3.2.1 基本概念…………………… 47
3.2.2 布尔模型…………………… 49
3.2.3 项权重……………………… 50
3.2.4 TF-IDF 权重 ……………… 52
3.2.5 文档长度归一化…………… 56
3.2.6 向量模型…………………… 57
3.2.7 概率模型…………………… 59
3.2.8 经典模型之间的简单比较 … 64
3.3 其他集合论模型………………… 64
3.3.1 基于集合的模型…………… 64
3.3.2 扩展布尔模型……………… 68

3.3.3 模糊集模型 …… 70
3.4 其他代数模型 …… 72
3.4.1 广义向量空间模型 …… 72
3.4.2 潜在语义索引模型 …… 74
3.4.3 神经网络模型 …… 75
3.5 其他概率模型 …… 76
3.5.1 BM25 模型 …… 77
3.5.2 语言模型 …… 78
3.5.3 随机差异模型 …… 83
3.5.4 贝叶斯网模型 …… 85
3.6 其他模型 …… 90
3.6.1 超文本模型 …… 90
3.6.2 基于 Web 的模型 …… 91
3.6.3 结构化文本检索 …… 91
3.6.4 多媒体检索 …… 92
3.6.5 企业和垂直搜索 …… 92
3.7 趋势和研究问题 …… 92
3.8 文献讨论 …… 93
第 4 章 检索评价 …… 96
4.1 介绍 …… 96
4.2 Cranfield 范式 …… 97
4.2.1 历史简述 …… 97
4.2.2 参考集 …… 98
4.3 检索指标 …… 98
4.3.1 精度和召回率 …… 98
4.3.2 单值总结：*P@n*，MAP，MRR，F …… 102
4.3.3 面向用户的指标 …… 105
4.3.4 折扣累积增益 …… 106
4.3.5 二元偏好 …… 109
4.3.6 排序相关性测度 …… 111
4.4 参考文档集 …… 115
4.4.1 TREC 参考集 …… 115
4.4.2 其他参考集 …… 121
4.4.3 其他小规模测试文档集 …… 121
4.5 基于用户的评价 …… 122
4.5.1 实验室中的人工实验 …… 122
4.5.2 并排面板 …… 122
4.5.3 A/B 测试 …… 123
4.5.4 众包 …… 124
4.5.5 使用点击数据的评价 …… 125
4.6 实践说明 …… 126
4.7 趋势和研究问题 …… 127
4.8 文献讨论 …… 127
第 5 章 相关反馈与查询扩展 …… 129
5.1 介绍 …… 129
5.2 反馈方法的框架 …… 129
5.3 显式相关反馈 …… 131
5.3.1 向量模型的相关反馈：Rocchio 方法 …… 131
5.3.2 概率模型的相关反馈 …… 133
5.3.3 相关反馈的评价 …… 134
5.4 基于点击的显式反馈 …… 134
5.4.1 眼动追踪和相关性评价 …… 134
5.4.2 用户行为 …… 135
5.4.3 点击作为用户偏好的指标 …… 136
5.5 通过局部分析的隐式反馈 …… 138
5.5.1 通过局部聚类的隐式反馈 …… 138
5.5.2 通过局部上下文分析的隐式反馈 …… 140
5.6 通过全局分析的隐式反馈 …… 141
5.6.1 基于相似度同义词典的查询扩展 …… 141
5.6.2 基于统计同义词典的查询扩展 …… 143
5.7 趋势和研究问题 …… 145
5.8 文献讨论 …… 145
第 6 章 文档：语言及属性 …… 147
6.1 介绍 …… 147
6.2 元数据 …… 148
6.3 文档格式 …… 149
6.3.1 文本 …… 149
6.3.2 多媒体 …… 149
6.3.3 图形和虚拟现实 …… 150
6.4 标记语言 …… 151
6.4.1 SGML …… 151
6.4.2 HTML …… 153
6.4.3 XML …… 155

6.4.4 RDF …… 157
6.4.5 HyTime …… 158
6.5 文本属性 …… 159
6.5.1 信息论 …… 159
6.5.2 自然语言建模 …… 159
6.5.3 文本相似度 …… 162
6.6 文档预处理 …… 163
6.6.1 文本的词汇分析 …… 163
6.6.2 去除禁用词 …… 164
6.6.3 词干提取 …… 165
6.6.4 关键词选择 …… 166
6.6.5 同义词典 …… 166
6.7 组织文档 …… 168
6.7.1 分类体系法 …… 168
6.7.2 分众分类法 …… 169
6.8 文本压缩 …… 170
6.8.1 基本概念 …… 170
6.8.2 统计方法 …… 171
6.8.3 统计方法：建模 …… 171
6.8.4 统计方法：编码 …… 173
6.8.5 字典方法 …… 179
6.8.6 压缩预处理 …… 180
6.8.7 文本压缩技术的比较 …… 181
6.8.8 结构化文本压缩 …… 182
6.9 趋势和研究问题 …… 183
6.10 文献讨论 …… 185
第7章 查询：语言及属性 …… 187
7.1 查询语言 …… 187
7.1.1 基于关键词的查询 …… 188
7.1.2 非关键词查询 …… 190
7.1.3 结构化查询 …… 192
7.1.4 查询协议 …… 194
7.2 查询属性 …… 195
7.2.1 Web查询的特征 …… 195
7.2.2 用户搜索行为 …… 197
7.2.3 查询意图 …… 197
7.2.4 查询主题 …… 199
7.2.5 查询会话与任务 …… 200
7.2.6 查询难度 …… 200
7.3 趋势和研究问题 …… 203
7.4 文献讨论 …… 204
第8章 文本分类 …… 205
8.1 介绍 …… 205
8.2 文本分类的特性描述 …… 206
8.2.1 机器学习 …… 206
8.2.2 文本分类问题 …… 206
8.2.3 文本分类算法 …… 207
8.3 无监督算法 …… 208
8.3.1 聚类 …… 208
8.3.2 朴素文本分类 …… 212
8.4 监督算法 …… 212
8.4.1 决策树 …… 214
8.4.2 k近邻分类器 …… 218
8.4.3 Rocchio分类器 …… 219
8.4.4 概率朴素贝叶斯文档分类 …… 221
8.4.5 支持向量机分类器 …… 224
8.4.6 集成分类器 …… 231
8.4.7 关于监督算法的结束语 …… 234
8.5 特征选择或降维 …… 234
8.5.1 项-类别出现列联表 …… 235
8.5.2 索引项文档频率 …… 236
8.5.3 TF-IDF权重 …… 236
8.5.4 互信息 …… 236
8.5.5 信息增益 …… 237
8.5.6 卡方检验 …… 237
8.5.7 特征选择的作用 …… 238
8.6 评价指标 …… 238
8.6.1 列联表 …… 238
8.6.2 准确率和错误率 …… 239
8.6.3 精度和召回率 …… 239
8.6.4 F测度和F_1 …… 240
8.6.5 交叉检验 …… 241
8.6.6 标准文档集 …… 241
8.7 类别组织——构建分类体系 …… 242
8.8 趋势和研究问题 …… 244
8.9 文献讨论 …… 244
第9章 索引和搜索 …… 247
9.1 介绍 …… 247

9.2 倒排索引 …… 249
9.2.1 基本概念 …… 249
9.2.2 完全倒排索引 …… 250
9.2.3 搜索 …… 252
9.2.4 排序 …… 256
9.2.5 构建 …… 257
9.2.6 压缩的倒排索引 …… 260
9.2.7 结构化查询 …… 261
9.3 签名文件 …… 262
9.4 后缀树和后缀数组 …… 264
9.4.1 结构：trie 树和后缀树 …… 265
9.4.2 简单字符串搜索 …… 266
9.4.3 复杂模式的搜索 …… 267
9.4.4 构建 …… 268
9.4.5 压缩的后缀数组 …… 270
9.5 序列搜索 …… 273
9.5.1 简单字符串：Horspool …… 274
9.5.2 复杂模式：自动机和位并行 …… 276
9.5.3 更快的位并行算法 …… 279
9.5.4 正则表达式 …… 281
9.5.5 多重模式 …… 282
9.5.6 近似搜索 …… 283
9.5.7 搜索压缩文本 …… 285
9.6 多维索引 …… 287
9.7 趋势和研究问题 …… 288
9.8 文献讨论 …… 289
第 10 章 并行与分布式信息检索 …… 293
10.1 介绍 …… 293
10.2 分布式信息检索系统的分类 …… 294
10.3 数据划分 …… 296
10.3.1 文档集划分 …… 297
10.3.2 文档集选择 …… 298
10.3.3 倒排索引划分 …… 299
10.3.4 划分其他索引 …… 302
10.4 并行信息检索 …… 303
10.4.1 介绍 …… 303
10.4.2 在 MIMD 架构上的并行信息检索 …… 305
10.4.3 在 SIMD 架构上的并行信息检索 …… 306
10.5 基于集群的信息检索 …… 310
10.6 分布式信息检索 …… 310
10.6.1 介绍 …… 310
10.6.2 索引 …… 313
10.6.3 查询处理 …… 315
10.6.4 Web 问题 …… 320
10.7 联合搜索 …… 320
10.8 在对等网络中的检索 …… 322
10.9 趋势和研究问题 …… 325
10.10 文献讨论 …… 326
第 11 章 Web 检索 …… 327
11.1 介绍 …… 327
11.2 一个有挑战性的问题 …… 328
11.3 Web …… 329
11.3.1 特性 …… 329
11.3.2 Web 图的结构 …… 331
11.3.3 对 Web 建模 …… 332
11.3.4 链接分析 …… 334
11.4 搜索引擎架构 …… 335
11.4.1 基本架构 …… 335
11.4.2 基于集群的架构 …… 336
11.4.3 缓存 …… 337
11.4.4 多级索引 …… 339
11.4.5 分布式架构 …… 340
11.5 搜索引擎排序 …… 342
11.5.1 排序信号 …… 342
11.5.2 基于链接的排序 …… 343
11.5.3 简单的排序函数 …… 345
11.5.4 排序学习 …… 345
11.5.5 学习排序函数 …… 346
11.5.6 质量评价 …… 347
11.5.7 Web 垃圾 …… 348
11.6 管理 Web 数据 …… 348
11.6.1 为文档分配标识符 …… 348
11.6.2 元数据 …… 349
11.6.3 压缩 Web 图 …… 349
11.6.4 处理重复数据 …… 349

11.7 搜索引擎用户交互 350
11.7.1 搜索矩形范式 351
11.7.2 搜索引擎结果页面 356
11.7.3 培养用户 363
11.8 浏览 364
11.8.1 扁平浏览 364
11.8.2 结构导向的浏览和 Web 目录 364
11.9 浏览之外 366
11.9.1 超文本和 Web 366
11.9.2 搜索与浏览相结合 366
11.9.3 Web 查询语言 367
11.9.4 动态搜索 367
11.10 相关问题 368
11.10.1 计算广告学 368
11.10.2 Web 挖掘 370
11.10.3 元搜索 371
11.11 趋势和研究问题 372
11.11.1 静态文本数据之外 372
11.11.2 目前的挑战 373
11.12 文献讨论 374
第 12 章 Web 爬取 376
12.1 介绍 376
12.2 网络爬虫的应用 377
12.2.1 通用 Web 搜索 377
12.2.2 聚焦爬取 378
12.2.3 Web 刻画 378
12.2.4 镜像 378
12.2.5 网站分析 379
12.3 爬虫的分类体系 379
12.4 架构和实现 380
12.4.1 爬虫架构 380
12.4.2 实际问题 382
12.4.3 并行爬取 384
12.5 调度算法 384
12.5.1 选择策略 385
12.5.2 重访问策略 387
12.5.3 友好策略 391
12.5.4 组合策略 393
12.6 评价 393
12.6.1 评价网络使用 393
12.6.2 评价长期调度 394
12.7 趋势和研究问题 395
12.7.1 爬取“暗网” 395
12.7.2 在网站帮助下的爬取 396
12.7.3 分布式爬取 396
12.8 文献讨论 396
第 13 章 结构化文本检索 398
13.1 介绍 398
13.2 结构化能力 399
13.2.1 显式和隐式结构对比 399
13.2.2 静态与动态结构对比 399
13.2.3 单一层次结构与多层次结构对比 400
13.3 早期文本检索模型 400
13.3.1 基于非覆盖列表的模型 401
13.3.2 基于相邻结点的模型 401
13.3.3 结构化文本结果排序 402
13.4 XML 检索 403
13.4.1 XML 检索中的挑战 403
13.4.2 索引策略 404
13.4.3 排序策略 405
13.4.4 去除重叠 412
13.5 XML 检索评价 413
13.5.1 文档集 414
13.5.2 主题 414
13.5.3 检索任务 415
13.5.4 相关性 416
13.5.5 测度 417
13.6 查询语言 419
13.6.1 特性 419
13.6.2 XML 查询语言分类 420
13.6.3 XML 查询语言样例 421
13.7 趋势和研究问题 425
13.8 文献讨论 427
第 14 章 多媒体信息检索 429
14.1 介绍 429
14.1.1 什么是多媒体 429

14.1.2 多媒体检索 429
14.1.3 文本检索与多媒体检索的对比 430
14.2 挑战 431
14.2.1 语义鸿沟 431
14.2.2 特征歧义性 432
14.2.3 机器生成的数据 432
14.3 基于内容的图像检索 433
14.3.1 基于颜色的检索 433
14.3.2 纹理 434
14.3.3 显著点 436
14.4 声音和音乐检索 437
14.4.1 指纹识别 437
14.4.2 语音识别 438
14.4.3 说话人识别 440
14.4.4 语音文档检索 440
14.4.5 音频基础知识 440
14.5 检索和浏览视频 443
14.5.1 视频摘要 443
14.5.2 静态摘要 444
14.5.3 图像拼接与跳跃剧照 445
14.5.4 动态摘要 446
14.5.5 交互式摘要 447
14.5.6 视觉与听觉浏览对比 448
14.5.7 摘要评价 448
14.6 融合模型：合并所有信息 449
14.6.1 人脸命名 449
14.6.2 图像命名 450
14.6.3 音频命名 451
14.6.4 结合音频与视频的音-视频语音识别 451
14.6.5 结合音频和视频的多媒体处理 453
14.7 分割 453
14.7.1 视频分割样例 454
14.7.2 视频分割方案 455
14.7.3 利用边缘的视频分割 455
14.7.4 语音分割 456
14.7.5 分割评价 457
14.8 压缩和 MPEG 标准 457
14.8.1 强度和采样 458
14.8.2 颜色 458
14.8.3 有损压缩 459
14.8.4 无损压缩 461
14.8.5 时间冗余 461
14.8.6 运动预测 461
14.8.7 MPEG 标准 462
14.9 趋势和研究问题 465
14.10 文献讨论 466
第 15 章 企业搜索 469
15.1 介绍 469
15.1.1 企业搜索的特点和应用 469
15.1.2 企业搜索软件 470
15.1.3 工作场所搜索 471
15.2 企业搜索任务 471
15.2.1 搜索支持任务的例子 471
15.2.2 搜索类型 473
15.2.3 研究企业搜索 473
15.3 企业搜索系统的结构 474
15.3.1 收集 474
15.3.2 提取 476
15.3.3 索引 477
15.3.4 文本注释的索引 477
15.3.5 查询处理 478
15.3.6 搜索结果的展示 479
15.3.7 安全模型 480
15.3.8 联合/元搜索 482
15.4 企业搜索评价 484
15.4.1 企业搜索的公开测试集 484
15.4.2 企业搜索内部评价 485
15.4.3 企业搜索调试 486
15.4.4 所能期待的是什么 487
15.5 不满意的可能原因 488
15.6 情境化和个性化 490
15.6.1 情境化的控制和工具 491
15.6.2 情境化：本地、企业或全球 493
15.6.3 轮廓的隐私 494

15.6.4 定义、建立和维护轮廓 … 494
15.6.5 用户建模 …… 495
15.6.6 隐式评价 …… 496
15.6.7 信息过滤 …… 496
15.6.8 社会化推荐系统 …… 497
15.7 趋势和研究问题 …… 497
15.8 文献讨论 …… 497
第 16 章 图书馆系统 …… 499
16.1 图书馆的信息环境 …… 499
16.2 联机公共检索目录 …… 500
16.2.1 OPAC 和书目记录 …… 501
16.2.2 来自 ILS 的信息检索 … 503
16.2.3 混合图书馆的整合 …… 504
16.2.4 OPAC 和最终用户 …… 505
16.2.5 ILS：供应商和产品 …… 506
16.3 信息检索系统与文档数据库 … 507
16.3.1 书目和全文数据库 …… 508
16.3.2 数据库记录的内容 …… 508
16.3.3 联机产业：数据库供应商 …… 510
16.3.4 来自文档数据库的信息检索 …… 511
16.4 组织机构内部的信息检索 …… 514
16.5 趋势和研究问题 …… 515
16.6 文献讨论 …… 516
第 17 章 数字图书馆 …… 517
17.1 介绍 …… 517
17.2 定义数字图书馆 …… 517
17.3 通用架构 …… 518
17.4 基本概念 …… 519
17.4.1 数字对象和馆藏 …… 519
17.4.2 元数据和目录 …… 520
17.4.3 资源库/档案库 …… 522
17.4.4 服务 …… 525
17.5 社会经济问题 …… 527
17.5.1 社会问题 …… 527
17.5.2 经济问题 …… 527
17.6 软件系统 …… 528
17.6.1 Greenstone …… 529
17.6.2 Eprints …… 529
17.6.3 DSpace …… 529
17.6.4 Fedora …… 529
17.6.5 ODL …… 530
17.6.6 5S 套件 …… 530
17.7 数字图书馆案例研究 …… 531
17.7.1 联网学位论文数字图书馆 … 531
17.7.2 国家科学数字图书馆 …… 532
17.7.3 ETANA-DL 考古数字图书馆 …… 532
17.8 趋势和研究问题 …… 532
17.8.1 评价 …… 532
17.8.2 集成 …… 533
17.8.3 其他研究挑战 …… 533
17.9 文献讨论 …… 534
附录 A 开源搜索引擎 …… 535
附录 B 作者简介 …… 549
参考文献 …… 554
索引 …… 654

引言

1.1 信息检索

信息检索（Information Retrieval，IR）是计算机科学的一大领域，主要研究如何为用户访问他们感兴趣的信息提供各种便利的手段，即：

> 信息检索涉及对文档、网页、联机目录、结构化和半结构化记录及多媒体对象等信息项的表示、存储、组织和访问。信息项的表示和组织必须便于用户访问他们感兴趣的信息。

在范围上，信息检索的发展已经远远超出了其早期目标，即对文档集进行索引并从中寻找有用的文档。如今，信息检索的研究包括建模、Web 搜索、文本分类、系统架构、用户界面、数据可视化、过滤和语言处理技术。

在研究方面，信息检索可以从两个相当不同和互补的视角展开研究：以计算机为中心的视角和以人为中心的视角。从以计算机为中心的视角来看，信息检索主要包括建立高效的索引，高性能地处理用户的查询，并开发排序算法以提高检索结果。从以人为中心的视角来看，信息检索主要包括研究用户的行为，理解他们的主要需求，并且相应地确定检索系统的组织和操作。鉴于前者在学术界和市场上的主导地位，本书主要论述以计算机为中心的视角。

1.1.1 信息检索的早期发展

5000 多年来，人类已经知道如何组织信息，为以后的检索和搜索服务。在最通常的形式，它一直是通过编辑、储存、组织和索引泥板、象形文字、纸草卷和书籍实现的。为存放 1
各种物品，人类还使用了特殊用途的建筑物，并称之为*图书馆*。表示图书馆的英语单词一个是“library”，来自拉丁文的“liber”，表示“书籍”；另一个是“bibliothek”，来自希腊文的“biblion”，表示“纸草卷”。

已知最古老的图书馆在公元前 3000—公元前 2500 年之间成立于厄尔巴。它位于“新月沃地”（Fertile Crescent），即目前的叙利亚北部。在公元前 7 世纪，亚述王亚述巴尼拔在底格里斯河（位于今日的伊拉克北部）建造了尼尼微图书馆，该图书馆在公元前 612 年，也就是被毁灭的那一年，共收藏 30 000 多块泥板。到了公元前 300 年，马其顿将军多利买梭特尔，在尼罗河口以马其顿国王亚历山大大帝（公元前 356—公元前 323 年）命名的亚历山大市，建造了亚历山大图书馆。700 年间，亚历山大图书馆和同城的其他图书馆一道，使得亚历山大成为西方世界的知识之都 [1164]。

从那时起，图书馆日渐扩大和繁荣，如今已遍布世界各地。它们构成了人类的集体记忆，并且越来越普遍。仅 2008 年，美国人去图书馆的次数就达到了约 13 亿次，借阅资料超过 20 亿件，并且这个数字每年增加的幅度都在 10%以上 [155]。

由于图书馆的信息容量一直在增长，因此有必要建立专门的数据结构——*索引*，进行快速搜索。不管采用哪种形式，索引都是每一个现代信息检索系统的核心。它们提供快速访问数据的方法以加快查询处理。我们将在第 9 章讨论索引技术。

数百年来，索引的形式都是手动建立的类目集。索引中的每个类目通常由标志相关主题的标签和指向相关文档的指针组成。虽然这些索引通常是由图书馆和信息科学的研究人员设计，但现代计算机的出现使得自动构建大规模索引成为可能，而这也加快了信息检索领域的发展。

信息检索的早期发展可以追溯到 20 世纪 50 年代 Hans Peter Luhn、Eugene Garfield、Philip Bagley 和 Calvin Moores 等开拓者所进行的研究工作，其中最后一位还发明了信息检索这个术语 [1692]。在 1955 年，Allen Kent 和他的同事发表了一篇论文，描述了精度(precision)⊖和召回率（recall)⊜两项评价指标 [903]，1962 年 Cyril Cleverdon 在所进行的 Cranfield 研究中沿用了它们 [394，395]。1963 年，Joseph Becker 和 Robert Hayes 出版了关于信息检索的第一部书籍 [164]。在 20 世纪 60 年代，Gerard Salton 和 Karen Sparck Jones 等人提出了现代信息检索中排序技术的基本概念，从而塑造了这一领域。1968 年，Salton 出版了他的第一部信息检索书籍。1971 年，N. Jardine 和 C. J. Van Rijsbergen 清晰地提出了“聚类假设”（cluster hypothesis） [827]。1978 年，第一届 ACM 信息检索会议（ACM Conference on IR，ACM SIGIR）在纽约州的罗切斯特举行。1979 年，C. J. Van Rijsbergen 出版了介绍概率检索模型的专著《Information Retrieval》[1624]。1983 年，Salton 和 McGill 出版了介绍向量检索模型的经典专著《Introduction to Modern Information Retrieval》[1414]。从那以后，信息检索研究群体日渐扩大，现在已包含来自世界各地成千上万的教授、研究人员、学生、工程师和从业人员。本领域最重要的会议——ACM 信息检索国际会议（ACM International Conference on Information Retrieval，ACM SIGIR)，现在每
2 年能吸引数百位参加者和数百篇投稿。

1.1.2 图书馆和数字图书馆中的信息检索

图书馆是采用信息检索系统搜寻信息的第一批机构。通常情况下，图书馆系统最初是由学术机构，后来由商业供应商开发。第一代图书馆系统是对现有流程的自动化，例如用作者姓名和书名检索卡片目录。第二代系统则增加了搜索功能，包括主题词和关键字的检索和查询操作。目前正在部署的第三代系统重点则是改进的图形界面、电子表单、超文本功能和开放式系统架构。

传统的图书馆管理系统供应商包括 Endeavor 信息系统公司、Innovative Interfaces 公司和 EOS 国际公司。在目前正在开发的研究系统中，值得关注的是位于加州大学的加州数字图书馆所开发的 MELVYL 系统，以及最初由加州大学伯克利分校开发、最近与利物浦大学合作的 Cheshire 系统。关于这些图书馆系统的进一步详情可参看第 16 章。

1.1.3 舞台中央的信息检索

虽然已经成熟，但直到最近，信息检索仍被视为只有图书管理员和信息专家感兴趣的狭窄领域。这种偏见已盛行多年，尽管多媒体和超文本的信息检索工具已经在现代个人计算机用户中迅速传播。万维网（World Wide Web，Web）在 20 世纪 90 年代初的引入彻底颠覆了所有这些看法。

1989 年蒂姆·伯纳斯-李发明的 Web，已成为人类知识和文化的万能信息库。它的成功

⊖ 在信息检索领域，“precision”也译为“查准率”。——译者注

⊜ 在信息检索领域，“recall”也译为“查全率”。——译者注

是基于对标准用户界面的构想——该界面不随计算环境的改变而改变，并允许任何用户创建自己的文件。利用 Web，数百万用户已经创造了数十亿的文档，从而构成了人类历史上最大的知识宝库。一个直接后果是，在 Web 上查找有用的信息并不总是一个简单的任务，通常需要提交查询给搜索引擎，即运行一个搜索任务，这完全就是信息检索技术。因此，几乎在一夜之间，信息检索与其他技术一起，站在了舞台的中央。

1.2 信息检索问题

现代信息检索系统的用户，例如搜索引擎用户，有多种多样的信息需求。在最简单的情况下，他们寻找指向企业、政府或者机构主页的链接；在稍微复杂的情况下，他们寻找完成工作任务或即时需求的信息。更复杂的*信息需求*（information need）则例如：

> *寻找所有与联邦政府在全国铁路运输公司*（National Railroad Transportation Corporation，AMTRAK）*融资中所扮演角色相关的文档*[⊖]。 3

用户需求的这种完整表示并不必然就构成提交给信息检索系统的最佳形式。相反地，用户可能首先要将此信息需求转换成*查询*（query）或者查询序列提交给系统。在其最常见的形式，这种转换总结了用户的信息需求，并产生一组关键字或索引项。给定用户查询，检索系统的主要目标是要获取有用或相关的信息并提交给用户。重点是*信息检索*，而不是*数据检索*。

为了有效地满足用户的信息需求，检索系统必须以某种方式“解释”信息项（即库中的文档）的内容，并根据和用户查询相关的程度对文档进行排序。文档内容的“解释”涉及从文档中提取文本的句法和语义信息并利用这些信息来匹配用户的信息需求。

> **信息检索问题**：*信息检索系统的主要目标是检出所有和用户查询相关的文档，并且把检出的不相关文档控制在最低限度。*

信息检索的困难在于不仅需要知道如何从文档中提取信息，而且还要知道如何用它来决定相关性。也就是说，*相关性*的概念对信息检索至关重要。

一个主要的问题是，对相关性的评估是个性化的，决定于被解决的任务及其上下文。例如，相关性可以随时间而改变（如新信息的出现），随位置而改变（例如，最相关的答案是距离最近的），甚至随设备而改变（例如，最好的答案是一篇简短的、容易下载和可视化的文档）。从这个意义上讲，不存在能在任何时间给任何用户提供完美答案的检索系统。

1.2.1 用户的任务

检索系统的用户必须把他们的信息需求转换成用系统提供的语言所描述的查询。利用信息检索系统，如搜索引擎，通常意味着指定一组词来传达信息的语义。我们称之为用户在*搜索*或*查询*他们感兴趣的信息。虽然搜索感兴趣的信息是 Web 检索的主要任务，但除了信息获取之外，搜索也可用于满足其他种类的用户需求，如购买商品和订位等，我们将在 1.4.3 节对此加以讨论。

现在考虑这样一种情况，用户的兴趣要么定义不清要么流于泛泛，以至于很难清晰地制定查询。例如，用户可能对关于赛车的一般信息感兴趣，可能会决定浏览与 F1 赛车、印地车赛和勒芒 24 小时耐力赛有关的文档。我们称这种情况为用户在*浏览*或者*导航*文档集中的

⊖ TREC 参考集的 168 主题。参看第 4 章。

文档，而不是搜索。它仍然是一个信息检索过程，但主要目标起初并不太清楚。这种情况下，任务更多的是探索式搜索，类似于对感兴趣信息的准序列搜索过程。

在这本书中，我们将检索系统的不同用户所进行的任务区分为两种截然不同的类型：搜索和浏览，如图 1-1 所示。第 2 章将详细介绍
4 这两种不同的任务。

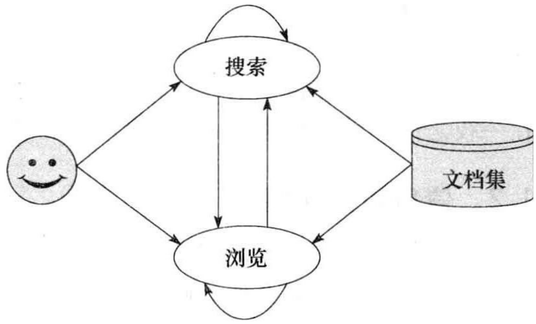


图 1-1 用户的任务

1.2.2 信息检索与数据检索

在信息检索系统的环境中，数据检索通常不足以满足用户的信息需求。数据检索主要包括确定集合中的哪些文档包含用户查询中的关键字。事实上，信息检索系统的用户更注重检出与某个主题相关的信息，而不是检索出符合用户查询的数据。例如，信息检索系统的用户愿意接受结果中包含查询项同义词的文档，即使这些文档没有包含任何查询项。也就是说，在一个信息检索系统中，检出的对象可以是不精确的，小错误可能被忽视。

与此相反，在数据检索系统中，1000 个检索对象中出现一个错误对象就意味着彻底失败。数据检索系统，如关系数据库，其处理对象具有明确定义的结构和语义；而信息检索系统处理的是没有很好结构的自然语言文本。数据检索能够为数据库系统的用户提供解决方案，但却不能解决检索与特定主题相关信息的问题。

1.3 信息检索系统

在本节中，我们提出信息检索系统软件架构的高层视图，并介绍响应用户查询的文档检索和排序过程。

1.3.1 信息检索系统的软件架构

为了描述信息检索系统，我们使用一个简单而通用的软件体系结构，如图 1-2 所示。建立信息检索系统的第一步是建立文档集，它可以是私有的，或者从 Web 上爬取。在第二种情况下，爬虫模块负责收集文档，我们将在第 12 章对此进行讨论。存储在磁盘上的文档集
5 通常称为*中央资源库*（central repository）。中央资源库里的文档需要进行索引，以进行快速检索和排序。最常用的索引结构是*倒排索引*（inverted index），它由文档集中所有不同的词组成，并为每个词建立一个包含这个词的文档列表。倒排索引将在第 9 章讨论。

文档集的索引建立之后，检索过程就可以启动。它既包括检索满足用户查询的文档，也包括点击超链接。在第一种情况下，我们说用户正在*搜索*感兴趣的信息；在第二种情况下，我们说用户在*浏览*感兴趣的信息。本节的其余部分介绍搜索。有关浏览的更详细的讨论，以及两种情况的比较，请参阅第 2 章。

为了进行搜索，用户首先指定一个反映他们信息需求的*查询*。接下来，对用户查询进行分析和扩展，例如加入查询词的拼写变体。扩展的查询，我们称之为*系统查询*，将与倒排索引进行匹配，并检索出一个文档子集。接下来，对文档子集排序并把排在最前面的文档返回给用户。换行排序的目的是找出最有可能被用户认为是相关的文档。这构成了信息检索系统中最关键的部分。正因为如此，第 3 章的信息检索模型介绍将非常详细，
6 且覆盖范围广泛。

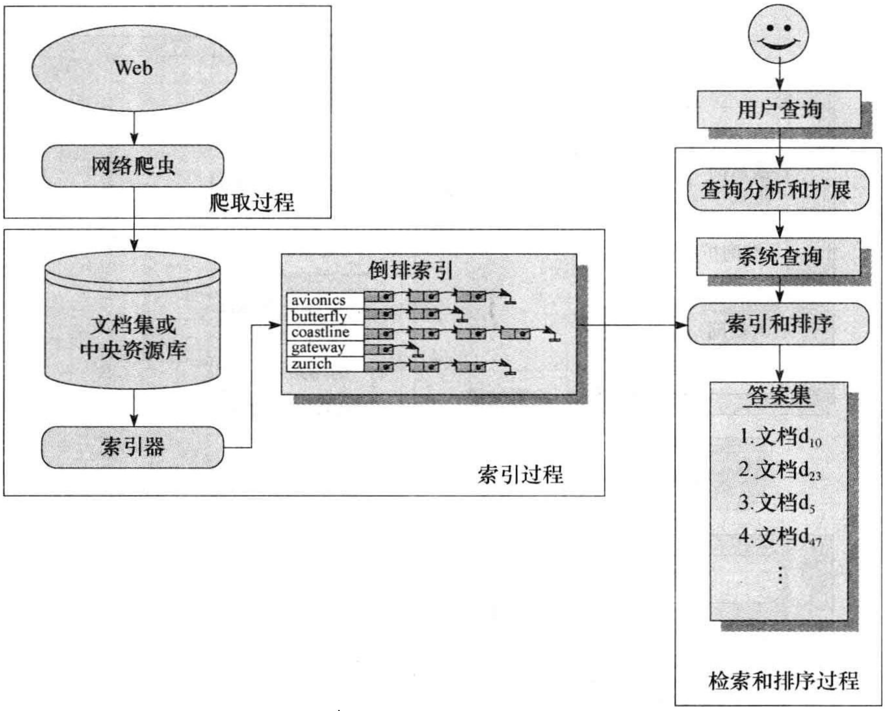


图 1-2 信息检索系统的高层软件架构。其中爬取是 Web 信息检索系统（如搜索引擎）额外要求的一个模块

鉴于判断相关性过程中所固有的主观性因素，评价答案集的质量是提高信息检索系统性能的关键步骤。系统的评价过程允许对排序算法进行微调以提高结果的质量，我们将在第 4 章对此加以讨论。最常见的评价过程是把信息检索系统产生的结果文档集和人类专家建议的结果进行比较。

为了提高排序的性能，我们可以收集用户的反馈，并使用这些信息来对结果重新排序。在 Web 中，最丰富的用户反馈形式是在返回结果上点击链接，我们将在第 5 章讨论。网页排序的另一个重要信息来源是页面间的超链接，可以从中发现权威度较高的页面，我们将在第 11 章讨论。

对一个完全成熟的信息检索系统（例如现代搜索引擎）而言，还有许多其他的概念和技术，其中大多数将在本书的其余章节内介绍。

1.3.2 检索和排序过程

为了描述检索和排序过程，我们对图 1-2 所示的模块进行进一步阐述，如图 1-3 所示。给定文档集中的文档，我们首先进行禁用词消除、词干提取等文本操作，并选择所有项的一个子集作为索引项，然后用索引项来构建文档的表示，这种表示可能比文档本身小（取决于选定的索引项子集）。

给定该文档表示，有必要建立一个文本索引。可以使用不同的索引结构，但最流行的是将在第 9 章讨论的*倒排索引*。生成所需索引的步骤就组成了*索引过程*，该过程必须在系统准备好处理任何查询之前离线执行。在索引过程中所耗费的资源（时间和存储空间）由检索系统在处理多次查询的过程中分摊。

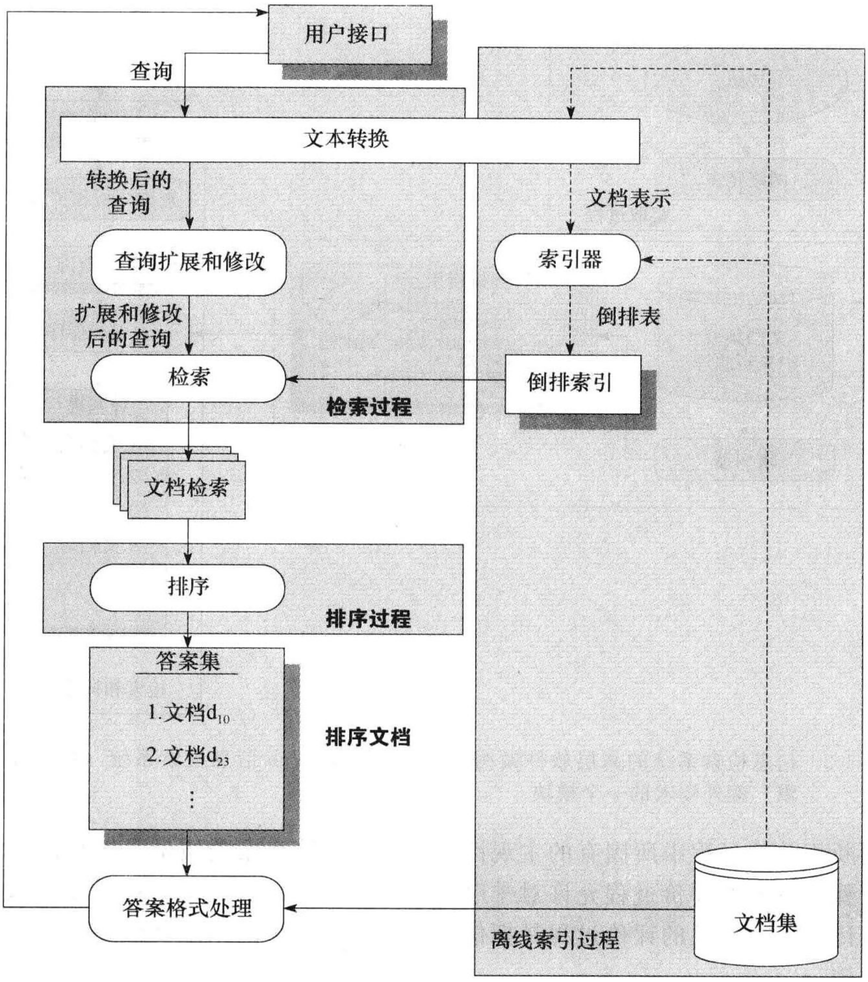


图 1-3 文档的索引、检索和排序过程

检索过程在给定文档集的索引之后启动。用户首先指定一个反映他们信息需求的*查询*，然后对此查询进行与文档类似的分析和修改操作。这里的典型操作包括适当的拼写校对和禁用词消除等。接下来，对转换后的查询进行扩展和修改。例如，系统可以对查询做出修改建议，并由用户确认。系统对扩展和修改后的查询进行处理，产生*检索文档集*，即由包含查询项的文档组成的集合。而先前建立的索引结构使得快速的查询处理成为可能。产生检索文档集的必要步骤就构成了*检索过程*。

接下来，检索出的文档将根据与用户需求的相关性进行*似然度*排序。由于用户所能感知的检索结果质量完全依赖于排序，因此这是最为关键的步骤。第 3 章我们将对排序过程进行详细介绍。系统将对排在最前面的文档进行格式处理并展现给用户。这些格式处理包括找出
7 文档的标题，根据查询项在文档中出现的上下文生成结果片段等。

1.4 Web

在本节中，我们讨论 Web 及其所揭示的电子出版时代。我们还会讨论 Web 如何改变搜索，也就是说，Web 对搜索任务的主要影响。最后，我们涵盖诸如安全和版权等由百万级的大规模 Web 用户而导致的实际问题。

1.4.1 Web 简史

在第二次世界大战结束时，美国总统富兰克林·罗斯福向后来获得高级政府职位的万尼瓦尔·布什咨询如何将在战争中掌握的技术应用到和平时期。布什首先做了名为“Science, 8
The Endless Frontier”（科学，无尽的前沿）的报告。该报告直接影响了美国国家科学基金会的建立。之后，他写了一篇影响深远的文章“As We May Think”（我们可以想象）[303]，讨论了可能在未来几年发明的新硬件和软件。用布什的话说，

> 百科全书将以全新的形式出现，资料之间由网络关联，随时可以放入扩展存储器（memex），并可以不断往里面添加新的信息 [303]。

“As We May Think”影响了许多人，包括 Douglas Engelbart。他在 1968 年 12 月的旧金山秋季联合计算机会议（Fall Joint Computer Conference）上运行了一个演示系统，推出了首个计算机鼠标、视频会议系统、远程会议系统和超文本。它是如此不可思议，以至于成为“所有演示之母”[1690]。演示中最让我们感兴趣的创新之处是*超文本*（hypertext）。该术语是由 Ted Nelson 在他的项目“世外桃源”（Xanadu）[1691] 中创造的。

超文本允许读者从一个电子文件跳转到另一个，这是蒂姆·伯纳斯-李（Tim Berners-Lee）在 1989 年所面临问题的一个重要属性。当时，伯纳斯-李在日内瓦的欧洲核子研究中心（Conseil Eouropéen pour la Recherche Nucléaire, CERN）工作。那里的研究人员如果想要与他人分享自己的文件就必须重新格式化文件，使其与内部的出版系统兼容 [803]。这很令人厌烦，产生了许多问题，其中许多问题需要由伯纳斯-李去解决。他意识到需要更好的解决方案。

欧洲核子研究中心碰巧是欧洲最大的因特网节点。伯纳斯-李认为，需要把共享的文件分散化，使得研究人员能够自由地分享他们的成果。他认为通过因特网链接的超文本将是一个很好的解决方案，并开始着手实现。1990 年，他写了 HTTP 协议，定义了 HTML 语言，编写了第一个 Web 浏览器——他称之为“万维网”，并搭建了第一个 Web 服务器。1991 年，他在因特网上发布了浏览器和服务器软件。Web 诞生了。

1.4.2 电子出版时代

Web 从一出现就取得了巨大的成功。现在网页的数量已远远超过 200 亿⊖ [487]，全世界的 Web 用户数也超过 17 亿 [815]。此外，众所周知在 Web 上有超过 1 万亿个不同的 URL[651]，即使其中许多是指向动态页面的指针，而不是静态的 HTML 页面。基于在线广告甚至实现了经济可持续发展的可行模式 [801]。

Web 的出现改变了这个世界，这一点很少有人能预见到。然而，人们想知道 Web 有哪些特性使得它如此成功，或者说，是否存在着某个单一的特性，对 Web 的成功起到决定性的作用？对这个问题的初步答案包括：简单的 HTML 标记语言、低成本的存取、因特网的 9
广泛普及、交互式的浏览器界面，以及搜索引擎。然而，虽然这些技术提供了基本的 Web 基础设施，但不是其流行的根源。那么特性是什么呢？

要强调这里我们提出的观点，让我们观察 200 年前某位作家的一生。

> 她在 1796 年和 1797 年之间完成了她的小说初稿。第一次投稿却被拒绝。因为最终失去了原稿，所以她在 1812 年改写了小说，并终于在 1813 年匿名出版，署名

⊖ 根据 http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html 上的博文，谷歌宣称已经搜集了超过 1 万亿个不同的 URL。

为“一位女士”[400]。

《Pride and Prejudice》（傲慢与偏见）是英国史上最受欢迎的三部书籍之一，另两部是《The Lord of the Rings》（指环王）和《Harry Potter》（哈利·波特）系列丛书。它先后被翻拍为 6 部电视剧和 5 部电影 [1694]。最近的一部由 Keira Knightley 和 Matthew Macfadyen 主演，获得了超过 1 亿美元的全球票房，Knightley 女士并因此获得了奥斯卡奖提名 [1693]。

简·奥斯汀一生所有的作品都是匿名发表。在整个 20 世纪，奥斯汀的小说从未绝版。各种版本的出现对《Pride and Prejudice》的普及功不可没。

Web 所带来的人际关系的根本性转变是出版自由。简·奥斯汀没有这种自由，所以她要不必须设法说服出版商相信作品的质量，要不就自己支付出版费用。由于无法支付，因此她不得不耐心等待了 15 年，直到出版商相信为止。

在 Web 世界中，这样的情况不会再发生。人们现在可以在 Web 上发布自己的想法，无须支付任何代价，也无须说服大型出版公司的编委会。而这样的想法经过一夜就会为数百万人所知晓。也就是说，Web 几乎完全解除了由大众传媒公司和自然地域壁垒所造成的限制，这导致了自由出版的新时代的诞生。我们称之为*电子出版时代*。

1.4.3 Web 如何改变搜索

Web 搜索是信息检索及其相关技术当今最突出的应用。事实上，任何搜索引擎的排序和索引组件在本质上都是信息检索技术。这个事实的一个直接后果就是，Web 对信息检索的发展已经产生了重大影响。

Web 对搜索的第一个重大影响与文档集自身的特点相关。Web 文档集是由分布在数百万个网站上的文档（或网页）组成，并通过超链接将页面上的文字块与其他网页连接。由于 Web 文档集所固有的分布式性质，因此在建立索引之前，需要收集所有文档的副本并将它们存储在一个中央资源库中。由 Web 所带来的这个信息检索过程中的新阶段称为*网页爬取*
10 (crawling)，将在第 12 章详细讨论。

Web 对搜索的第二个重大影响与文档集的大小以及每天提交的用户查询数量有关。Web 比以往任何已知的文档集更大，且增长速度更快，以至于现在的搜索引擎所需要处理的文本数量已经远远超过 200 亿页 [487]，远大于以往的任何文档集。此外，虽然对于用户查询的数量有各种规模的估计，但都认为比以往任何时候都多。海量文档集和海量查询流量的结合，使得搜索引擎的性能和可扩展性要求大大超过以往任何信息检索系统 [151]。也就是说，性能和可扩展性已成为 Web 信息检索系统的重要特性，其重要程度远远超过了它们在以往检索系统中的地位。虽然本书不讨论搜索引擎的性能和可扩展性问题，但是读者可以参考第 11 章关于本主题的文献（见文献讨论章节）。

Web 对搜索的第三个重大影响也与海量文档集有关。在非常大的文档集中预测相关性比以前更难。基本上，任何查询都会检索出很多匹配查询项的文档，这意味着检索文档集中有许多*噪声*。也就是说，检索文档集中的大部分文档似乎与查询相关，但实际上据大多数用户判断却是不相关的。此问题首次出现于早期的 Web 搜索引擎中，并随着 Web 的增长变得更严重。幸运的是，Web 还提供了标准文档集所没有的、缓解上述问题的新证据来源，如超链接、用户在结果文档中的点击行为等。在第 11 章中，我们将讨论 Web 上的相关性预测问题。

Web 对搜索的另外两个主要的影响源于这样的事实：Web 已不再仅仅是文档和数据库，也是一个商业媒介。直接的含义就是，搜索问题已经超出了对文字资料的寻找，还扩展到其

他用户需求，例如查询一本书的价格、酒店的电话号码、下载软件的链接。对这些类型的信息，提供有效的答案经常需要确定和关注对象相关联的一些结构化数据，如价格、地点，或主要特性描述等。这些新的查询类型将在第 7 章讨论。

Web 对搜索的第五个、也是最后的影响来自于 Web 广告和其他经济激励。作为大众化互动媒体，Web 的持续成功创造了广告和电子商务等形式的经济开发激励机制。这些激励措施也导致了 Web 垃圾信息的泛滥，也就是把商业信息伪装成纯粹的信息内容。Web 上的垃圾信息越来越普遍，有时是如此引人注目，并且与真正的相关内容相混淆，使得寻找相关信息甚至比以前更困难。正因为如此，认为垃圾内容使得相关性变差，也就是说垃圾信息的存在使得现有的排序算法产生的答案比没有垃圾内容的情况差很多，这也不是一点道理都没有。这种困难是如此之大，以至于现在需要谈论敌对 Web 检索，我们将在第 11 章对此加以讨论。 11

1.4.4 Web 上的实际问题

电子商务是当今 Web 一个惠及亿万人民的大趋势。在电子交易中，买方通常提交信用资料给供应商进行收费。信用资料最常见的形式就是信用卡号。出于安全原因，这些信息通常是加密的，由机构和公司部署的验证过程自动完成。

除了安全外，另一个引起关注的主要问题是隐私。通常，只要不被公开，人们都愿意交换信息。原因有很多，但最常见的是防止由第三方滥用自己的私人信息。因此，隐私是另一个影响 Web 的部署却并没有得到妥善解决的问题。

另外两个重要的问题是著作权和专利权。Web 数据的广泛分布如何影响各个国家的版权和专利法，目前还很不明朗。这一点很重要，因为它影响了建立和部署大型数字图书馆的业务。举例来说，网站是否要像出版商一样监督发布的所有信息？如果是的话，如果发布的信息被滥用，它是否需要负责（即使它不是信息源）？

此外，其他值得关注的实际问题包括扫描、光学字符识别（Optical Character Recognition，OCR），以及跨语言检索（用一种语言提交查询，但检索出的文档是另一种语言）。但是，本书将不会对这些实际问题进行详细介绍，因为它不是我们的主要关注点。有兴趣的读者可以参考 Lesk 的著作 [1005]。

1.5 本书的组织结构

1.5.1 本书的重点

虽然信息检索越来越引起人们的兴趣，但广泛覆盖本领域众多主题的现代信息检索教科书仍很难找到。本书从计算机科学家的视角出发，介绍信息检索领域的整体研究现状，试图部分地填补这一鸿沟。这意味着本书的关注点是信息检索系统所使用的计算机算法和技术。图书馆专家和信息科学研究人员的视角则截然不同，他们从以用户为中心的角度解释信息检索系统，其关注点不是如何自动地结构化、存储和检索信息，而是试图理解人们如何解释和使用信息。虽然本书的大部分章节专注于从计算机科学家的视角研究信息检索系统，但在本书的用户界面部分和最后两章的部分章节依然讨论了以人为中心的视角。

本书着重强调与信息检索紧密相关的不同领域需要整合在一起。因此，除了覆盖文本检索、图书馆系统、用户界面和 Web 之外，本书也介绍了可视化、多媒体信息检索和数字图书馆。

虽然有多位专家撰写了部分章节，本书依然是一本教科书，其内容和结构由两个主要作者进行了精心设计，他们也撰写或合写了全书 17 章中的 12 章。此外，所有其他作者撰写的 12

章节都已审慎修改、编辑，并被整合进入统一的框架。该框架规定了结构一致性、统一风格、共同词汇表、共同书目，以及适当的交叉引用。在每章的结尾讨论了研究问题、趋势和参考文献。这种讨论对研究生以及研究人员应该是有价值的。

1.5.2 本书的内容

由于信息检索是有五十多年历史的学科，一本书只能涵盖本领域全部知识的有限部分。尽管如此，为了获得对信息检索技术广泛深入的理解，仍然需要了解一些核心的关键概念、方法和技术。为了尽量覆盖这些概念和技术，我们撰写了 17 章内容，构成了本书第 2 版。由于第 1 版是十多年前出版的，因此本书第 2 版的所有章节和第 1 版截然不同。事实上，一半以上的素材是新的或已重写，目的或者是为了全面覆盖最新的研究结果，或者是为了简化符号，或者是为了介绍第 1 版尚未涉及的相关主题。为了说明这一点，本书增加了文本分类、结构化文本检索、Web 爬取，以及企业搜索等章节。此外，对相关反馈、多媒体、Web、图书馆系统、数字图书馆、检索评价和建模的章节已进行了大量修改和更新。

图 1-4 说明了本书的组织结构。本章介绍信息检索问题、Web 的简史，并分析其对信息检索的影响。因为搜索已经成为信息检索技术的主要应用领域，所以第 2 章论述用户搜索界面的设计。第 2 章是全新的，和第 1 版的用户界面章节截然不同，旨在为读者理解信息检索问题提供一个自顶向下的视角。

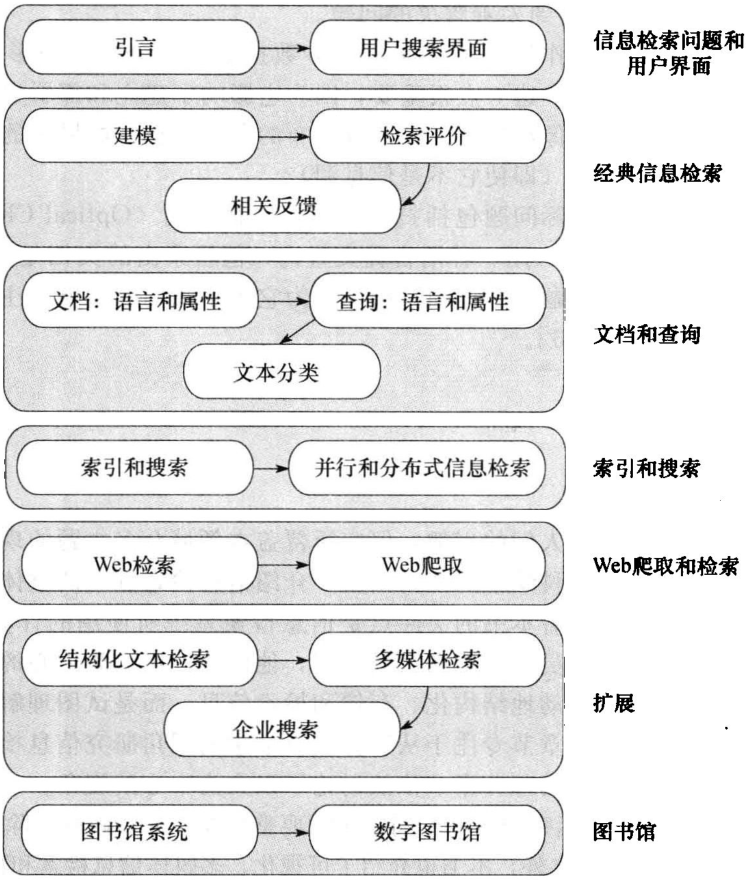


图 1-4 本书的组织结构

接下来的三章涉及经典的信息检索模型，包括排序模型、结果质量评价和用户相关反馈。这三章超过一半的素材是新的，证明了信息检索在过去 10 年的快速发展。我们的讨论广泛而深入。第 3 章讨论 14 个不同的信息检索模型，这些模型旨在对答案集中的每个文档打分并按照分数排序。我们从经典的布尔、向量和概率模型开始，然后对这 3 个经典模型中的每一个引入 3 个变种，包括基于集合的模型、广义向量模型、BM25 和语言模型。第 4 章就信息检索系统的结果质量评价，探讨了许多技术细节。我们首先进行了简单的历史回顾，包括 Cleverdon 针对索引系统评价的开创性工作，及其如何发展成为 Cranfield 范式。接下来介绍精度-召回率关系图，针对相关性分级的折扣累积增益（Discounted Cumulative Gain，DCG）指标，以及针对不完全相关性评价的二元偏好（Bpref）指标。我们同时也讨论了斯皮尔曼系数和肯德尔等级相关系数等排序关联度指标，并详尽介绍了 TREC 文档集和其他众多的小测试文档集。最后，讨论 Web 特有的评价方法，例如并排面板（side-by-side panels），并讨论如何将点击数据解释为相关性指标的方法。第 5 章讨论用户进行相关反馈的隐式和显式方法，以及如何使用它们来改变最终的排序。这些方法和查询扩展技术直 13
接交织在一起。这三章涵盖了所有经典的信息检索基本概念，即解决信息检索问题并评价结果的技术和方法。

接下来的三章讨论文档和查询的概念和相关技术，以及如何通过文本分类组织文档和查询。第 6 章讨论文本属性，例如词汇在文档的分布、自然语言模型、SGML、HTML 和 XML 等标注语言、文本处理与分析，以及压缩方法。第 7 章讨论各种查询属性，包括查询关键词的分布、Web 查询的特点，以及基于关键词的查询语言、结构化形式和查询协议等。第 8 章讨论组织文档和查询的算法和方法。我们讨论的重点是文档分类，因为这是最常见的情况。我们区分无监督和监督的文本分类算法。对于无监督的方法，我们介绍文本聚类算法，如 K 均值算法及其变种。对于监督方法，我们讨论 6 种不同类型的文本分类算法，即 14
决策树、最近邻、Rocchio、朴素贝叶斯、支持向量机和集成分类器。我们还会详细讨论如何评价分类结果。鉴于文本分类如今是信息检索的一项关键技术，所以本章是全新的，也是本书和第 1 版的重要区别之一。

再接下来的两章讨论索引和搜索文档集使用的技术。第 9 章讨论各种索引和搜索技术，包括序列搜索、倒排索引和后缀数组。我们还介绍索引压缩技术，以及如何使用它们来提高检索速度。第 10 章讨论并行和分布式索引以及（查询）搜索过程的体系结构和算法。提交给搜索引擎的海量查询只能由分布式计算机集群处理，这是现代 Web 的主要趋势。

之后的两章覆盖 Web 文档的爬取、检索和排序。第 11 章讨论 Web 检索，介绍 Web 的属性，搜索引擎的体系结构，HITS 和 Page rank 等链接分析算法，以及 Web 文档排序。虽然本章没有包括该领域的全部研究——当然任何一章都不可能，但它的确说明了搜索引擎如何获益于信息检索算法和技术。第 12 章讨论 Web 爬取。我们首先简要回顾了 Web 爬取技术的发展历史，然后讨论 Web 爬取的架构和实施问题。接下来是调度算法，这是任何爬取算法的核心部分，它确定下一步应该爬取哪些网页。最后，我们讨论 Web 爬取的评价过程。

Web 搜索的扩展包括结构化文本检索和多媒体检索，这两个是与 Web 日益相关的主要领域，另外还包括企业搜索。第 13 章讨论结构化文本检索，这是全新的章节，反映了自本书第 1 版出版以来该领域的迅速发展。其中包括早期的文本检索模型，XML 的索引和排序模型，XML 检索的评价方法和 XML 查询语言。第 14 章的多媒体信息检索也是全新的内容，从信息检索视角出发，从自顶向下的角度讨论多媒体检索。涵盖的内容包括基于内容的图像检索、音频和音乐检索，以及视频检索。将基于内容的图像检索、音频、音乐和视频检

索组合成一个单一的搜索机制需要融合模型，我们也会对此进行讨论。最后介绍 MPEG 标准。第 15 章讨论在机构和企业内部检索信息的企业搜索系统，包括它们和 Web 搜索系统的区别，以及在设计和实现方面的挑战。

本书的最后两章包括图书馆系统和数字图书馆。第 16 章讨论商业文档数据库、集成图书馆系统（Integrated Library System，ILS）和联机公共检索目录。商业文档数据库仍是当今最大的信息检索系统。例如 LEXIS-NEXIS 有一个由超过十亿的文档组成的数据库，每年提供数百万次查询。第 17 章已全面改写，提供对数字图书馆最新技术和趋势的详尽描述。首先是历史概述，随后讨论基本概念、社会和经济问题，以及 7 个独特的数字图书馆系统。最后，我们还讨论了数字图书馆中的重要个案，例如学位论文网络数字图书馆（networked digital library of theses and dissertations）、国家科学数字图书馆（national science digital li-
15 brary）和 ETANA 考古数字图书馆（ETANA archaeological digital library）。

本书还包括两个附录。附录 A 评论 27 个开源搜索系统，包括 HtDig、Indri、Lucene、MG4J、Omega、Omnifind、SwishE、Swish＋＋、Terrier 和 Zettair。附录 A 对这些搜索系统从索引构建时间、查询处理性能和存储需求等方面进行比较分析。附录 B 是对本书有贡献的所有作者的简介。最后是全书所使用的 1800 多篇参考文献。

虽然本书第 2 版的大部分资料还是纯粹的教科书风格，但我们还是为对研究有兴趣的读者增加了更多的内嵌引用。虽然我们试图平衡内容的广度和深度，但由于我们自己的专长和研究兴趣，有一些题目论述得更详细些。如果我们错过了一些主题或重要的细节或引用，我们在此提前道歉。

从经典的信息检索到 Web，从信息组织算法到现代数字图书馆，从搜索引擎所使用的索引和搜索技术到结构化文本搜索、多媒体搜索等需要扩充的新技术，本书第 2 版旨在从一个广泛而深入的视角论述信息检索的概念和技术，这些技术在搜索引擎中的应用，以及对相关领域（如信息科学、多媒体、数据库和数字图书馆）知识的影响。

1.6 本书的教学资源网站

本书的网站是 http://www.mir2ed.org，其中包含全书所有章节的幻灯片，可以作为教学资源使用。除幻灯片之外，也包括了词汇表、练习题和对面向不同听众的不同课程的详细教学建议，例如：

- 信息检索，计算机科学专业，本科生水平；
- 高级信息检索，计算机科学专业，研究生水平；
- 多媒体检索，计算机科学专业，本科生水平；
- 信息检索，信息系统专业，本科生水平；
- 信息检索，图书馆学专业，本科生水平；
- Web 检索，通识教育，本科或研究生水平；
- 数字图书馆，通识教育，本科或研究生水平。

此外，网站提供一个参考文档集供实验之用，包含 1239 篇来自 Cystic Fibrosis 参考集的文档，100 个信息需求和详尽的相关性评价数据 [1454]。而且，网站包括连接不同大学的信息检索课程、研究组、出版机构以及与信息检索及本书相关的其他资源的链接。

16 最后，本书网站还将公开发布与本书相关的重要新成果和补充信息，以及勘误表。

1.7 文献讨论

现在市面上已经有许多关于信息检索的其他书籍，由于目前对该主题的广泛兴趣，最近

也出现了一些新书。下面我们将本书与这些之前出版的书进行简单的比较。

信息检索领域的经典参考书是 van Rijsbergen 的《Information Retrieval》[1624]（网上可以找到），以及 Salton 和 McGill 的《Introduction to Modern Information Retrieval》[1414]。本书对于数据和信息检索的区别借鉴了前者，对于信息检索过程的定义则受到了后者的影响。然而，25 年过去了，这两本书现已过时，不能涵盖信息检索的新进展。

其他三本众所周知的信息检索著作是 Frakes 和 Baeza-Yates 编辑的《Information Retrieval：Data Structures & Algorithms》[582]，Witten、Moffat 和 Bell 撰写的《Managing Gigabytes-Compressing and Indexing Documents and Images》[1709]，以及 Lesk 的《Practical Digital Libraries；Books，Bytes，& Bucks》[1005]。这三本书都和本书互为补充。第一本偏重信息检索的数据结构和算法，有助于迅速实现已知算法的原型。第二本偏重索引和压缩技术，同时除了文本之外，也覆盖图像。本书由此借鉴了文本化图像的概念。第三本偏重数字图书馆及其实际问题，例如历史、分布、可用性、经济意义和知识产权。关于经典信息检索较新的著作包括 Hersh 的 [749]，Chowdhury 的《Introduction to Modern Information Retrieval》第 3 版 [382]。这两本著作的视角都比本书窄。Meadow、Boyce、Kraft 和 Barry 的《Text Information Retrieval Systems》第 3 版 [1112] 着重介绍信息及其表示。Allen 的《Information Tasks：Toward a User-Centered Approach to Information Systems》[32] 是关于信息系统的一般性著作，它采用以用户为中心，而不是以计算机为中心的视角阐述检索。从信息搜寻的视角，则需要提及 Marchionini 的《Information Seeking in Electronic Environments》[1082]，以及 Tedd 和 Hartley 的《Information Seeking in The Online Age：Principles and Practice》[977]。

某些章节有补充书籍。例如，许多书籍讨论信息检索和超文本，包括 Agosti 和 Smeaton 编辑的《Information retrieval and hypertext》[20]。多媒体检索也是如此，例如 Steinmetz 和 Nahrstedt 的《Multimedia-Computing，Communications and Applications》[1534]，以及 Alessi 和 Trollip 的《Multimedia for learning：methods and development》[25]。Hersh 的《Information Retrieval-A Health and Biomedical Perspective》[749] 是一本有趣的书，从健康和生物医药角度讨论信息检索。虽然标题中没有信息检索，但 Rosenfeld 和 Morville 的《Information Architecture for the World Wide Web：Designing Large-Scale Web Sites》第 3 版 [1157] 介绍了 Web 上的信息架构，是本书第 11 章的有益补充。Menasce 和 Almeida 的《Capacity Planning for Web Performance：Metrics，Models，and Methods》[1118] 阐述了如何利用排队论来预测 Web 服务器的行为。Chakrabarti 的《Mining the Web：Discovering Knowledge from Hypertext Data》[349] 介绍了 Web 知识挖掘的方法。此外，还有许多书籍说明如何从 Web 发现信息，如何使用搜索引擎。

Sparck Jones 和 Willet 编辑的《Readings in Information Retrieval》[1510]，与其说是一本合著，不如说是论文集。本书具有连贯性和广泛性，是更合适的学科教材。不过，该论文集仍然是有价值的研究工具书。Grefenstette 编辑的《Cross-Language Information Retrieval》是一本与跨语言信息检索有关的论文集 [674]。读者如对这个特定主题感兴趣，那么这本论文集就是本书很好的补充。此外，Maybury 编辑的《Intelligent Multimedia Information Retrieval》是一本偏重智能多媒体检索的论文集 [1101]，而 Strzalkowski 编辑的《Natural Language Information Retrieval》则关注自然语言信息检索 [1538]。为了纪念 Karen Sparck Jones，Tait 编辑的《Charting a New Course：Natural Language Processing and Information Retrieval》讨论自然语言处理与信息检索的关系 [1554]。其他一些合著探讨了 17

信息检索和不确定性与逻辑的关系 [444]、语言模型 [453]、认知检索 [1515] 和 TREC 评价 [1654]。

Korfhage 的《Information Storage and Retrieval》[931] 覆盖的材料比本书少很多，且不够具体。例如，该书没有详细讨论数字图书馆、Web、多媒体，以及并行处理。类似地，Kowalski 和 Maybury 的《Information Storage and Retrieval Systems: Theory and Implementation》第 2 版 [937]，还有 Shapiro 等人的《Automated Information Retrieval: Theory and Text-Only Methods》[1453] 都没有详细介绍这些内容，且定位也不同。Grossman 和 Frieder 的《Information Retrieval: Algorithms and Heuristics》 [682] 没有讨论 Web、数字图书馆和可视化界面。Berry 和 Browne 的《Understanding Search Engines-Mathematical Modeling and Text Retrieval》 [194] 是一本在搜索引擎语境下讨论经典信息检索的著作。其他一些专著则分别偏重于信息检索的数学基础 [505]、检索的几何解释 [1625]，以及标记结构在信息检索中的智能应用等 [942]。

近期 Ingwersen 和 Jarvelin 关于信息搜寻的著作《The Turn: Integration of Information Seeking and Retrieval in Context》[810] 力图从延伸的认知角度，而非基于 Cranfield 范式的实验模型，来解释信息检索。这直接影响了系统的评价方法。采用认知角度阐述信息检索，但却专注于搜索引擎的另一本专著是 Belew 的《Finding Out About: A Cognitive Perspective on Search Engine Technology and the WWW》[170]。最近的一本探索性搜索著作是 White 和 Roth 的《Exploratory Search: Beyond the Query-Response Paradigm》[1686]。

最近，Manning、Raghavan 和 Schutze 撰写了一部介绍经典信息检索和 Web 搜索的著作《Introduction to Information Retrieval》[1081]，该书的视角相当连贯而优雅，着重于基本概念，因此没有像本书一样探讨搜索界面等问题，也没有详尽介绍各种信息检索模型。此外，该书对于检索质量的评价和 Web 爬取介绍得很少，并且没有介绍结构化文本检索、多媒体检索、图书馆搜索系统和数字图书馆等内容。

Croft、Metzler 和 Strohman 的《Search Engines-Information Retrieval in Practice》是一本更新的著作 [449]，着重介绍搜索引擎，可作为本科生教材。该书提供的材料可用于讲述如何应用信息检索技术搭建搜索引擎的介绍性课程，因此其覆盖范围比本书窄，且没有包含搜索界面、相关反馈、查询扩展、多媒体和图书馆等材料。另外，对于建模、检索评价、文本分类和结构化文本检索的材料也不如本书详尽深入。

最后，几乎和本书同时出版的是 Büttcher、Clarke 和 Cormack 的《Information Retrieval: Implementing and Evaluating Search Engines》[304]。该书偏重信息检索系统的实现和评价，内容包括 XML 检索、并行搜索引擎和 Web 搜索。

对关注研究结果的读者而言，讨论信息检索及相关主题的学术期刊主要包括：

- 《Journal of the American Society of Information Sciences and Technology》 (JASIST, Wiley and Sons)
- 《ACM Transactions on Information Systems》(TOIS)
- 《Information Retrieval》(Kluwer)
- 《Information Processing and Management》(IP&M, Elsevier)
- 《ACM Transactions on the Web》
- 《IEEE Transactions on Knowledge and Data Engineering》(TKDE)
- 18 《Information Systems》(Elsevier)
- 《Knowledge and Information Systems》(KAIS, Springer)

- 《Data and Knowledge Engineering》(DKE, Springer)
- 《D-Lib Magazine》
- 《International Journal on Digital Libraries》(Springer)

与信息检索最相关的会议包括：

- ACM SIGIR International Conference on Information Retrieval (ACM SIGIR 信息检索国际会议)
- ACM International Conference on Web Search and Data Mining (WSDM) (ACM Web 搜索和数据挖掘国际会议)
- World Wide Web Conference (WWW), search track (万维网会议搜索分会)
- ACM Conference on Information Knowledge and Management (CIKM) (ACM 信息与知识管理会议)
- European Conference on IR (ECIR) (欧洲信息检索会议)
- String Processing and Information Retrieval Symposium (SPIRE) (国际字符串处理和信息检索会议)
- Text REtrieval Conference (TREC) (文本检索会议)
- INitiative for the Evaluation of XML retrieval (INEX) (INEX XML 检索评测)
- Cross Language Evaluation Forum (CLEF) (跨语言评价论坛)
- International Conference on Multimedia Retrieval (ICMR) (国际多媒体检索会议)，该会议新近由 ACM MIR 和 CIVR 合并而成
- Joint ACM-IEEE Conference on Digital Libraries (JCDL) (ACM-IEEE 数字图书馆联合会议)
- European Conference on Digital Libraries (ECDL) (欧洲数字图书馆会议) 19

用户搜索界面

——Marti Hearst 著

2.1 介绍

本书大部分的内容描述搜索引擎和信息检索系统的算法。而本章关注的是搜索系统的用户和搜索系统所显示的视窗：*用户搜索界面*。用户搜索界面的作用是帮助用户理解和表达他们的信息需求，并帮助用户制定他们的查询，在可用的信息源中进行选择，理解搜索结果，以及跟踪他们的搜索进程。

本书的第1版很少提到有关如何建立有效搜索界面的问题。在这些年间，人们开始逐步认识到哪些想法是有效的，哪些是无效的。本章简要总结了一些在学术研究以及商业系统部署方面最先进的搜索界面设计方法，讨论人们是如何搜索的、现今的搜索界面、搜索界面的可视化以及对用户搜索界面的评价。

2.2 人们如何搜索

搜索任务的范围可以从相对简单的（例如，查找有争议的事实或者查找天气信息）到丰富而复杂的（例如，求职和规划假期）。搜索界面应该支持一定范围内的任务，同时也要考虑到人们希望如何寻找到他们想要的信息。本节总结了与在线信息搜寻过程相关的理论模型
21 和经验观察。

2.2.1 信息查找与探索式搜索

与搜索界面交互的不同方式取决于任务的类型、搜索过程中投入的时间和精力，以及信息搜寻者的专业知识。Web搜索引擎中所使用的简单的交互式对话最适合寻找问题的答案、搜索网站，或者作为搜索的起点寻找其他资源。但是，正如Marchionini [1085] 指出的那样，搜索引擎“依次接收信息”的界面本身有局限性，在许多情况下正在被专业搜索引擎所取代——如对于旅游和健康信息的搜索，专业引擎能够提供更丰富的互动模式。

Marchionini [1085] 给出了*信息查找*（information lookup）和*探索式搜索*（exploratory search）的区别。信息查找任务类似于事实检索或问题回答，只需要简短而离散的信息即可：数字、日期、名称或文件和网站的名称。标准的Web搜索（以及标准数据库管理系统查询）在这些方面可以做得很好。

Marchionini将信息搜寻任务中的探索式搜索的类别划分为*学习*和*调查*两类。学习搜索需要多个查询响应对，并需要用户花费时间扫描和读取多个信息项，并综合这些内容来形成新的理解。调查指的是一个更长期的过程，意指“在相对较长的一段时间内进行多次迭代，返回的结果可能要在整合进个人和专业知识库之前，进行严格的评估。”[1085] 调查搜索可能被用做辅助计划安排、发现知识鸿沟，或者监视一个持续性的话题。有些种类的调查搜索关注于发现全体或大部分的可用相关信息（高召回率），比如诉讼研究或学术研究等。

其他人的一些工作支持了这种观点，O'Day 和 Jeffries[1219] 在研究了那些反复出现的深度复杂信息需求之后（他们主要关注于商业智能领域），发现信息搜寻过程是由一系列相

互关联但又不完全相同的搜索所组成的。他们还发现，一个搜索目标的结果往往会引发新的目标，从而引发新的搜索方向，但问题的背景和先前的搜索会从搜索的前一个阶段延续到下一阶段。他们还发现，搜索所带来的主要价值体现在搜索过程中持续的学习和所获得的信息，而不只是最后的搜索结果。

更广泛地说，信息搜寻可以被看做是更大过程当中的一部分，正如文献［1272，1401，1400］提到的*意义建构*（sensemaking）那样。意义建构是一个迭代的过程，它从一个大的信息集合中制订出一套概念表示方法。Russell 等人［1401］观察到，在意义建构中，主要的工作都致力于如何把好的表示方法、思考形式，以及面临的问题结合起来。他们描述了为给定的任务制定和明确其中的重要概念的过程。搜索只是这一过程中的一个部分；有些意义建构过程可能自始至终都需要搜索的参与，而另一些则是先进行一组搜索，然后再进行一系列的分析和综合。那些深层的分析任务需要进行意义构建，同时伴随着搜索，例如法律发现过程、流行病学（疾病跟踪）、通过研究顾客投诉来改善服务并获取商业智能等。 22

2.2.2 信息搜寻的经典模型与动态模型

研究人员已经构造出很多关于人们如何搜索的理论模型。Sutcliffe 和 Ennis［1547］提出的信息搜寻过程的经典模型将其定义为由 4 个主要活动所构成的周期性过程：

- 明确问题
- 表达信息需求
- 构造查询
- 评价结果

信息搜寻过程的标准模型包含一个潜在的假设，即用户的信息需求是静态的，信息搜寻的过程是一个对于查询项进行连续提炼的过程，直到所有且仅有那些与原始信息有关的文档被检索出来为止。最近的模型强调了搜索过程的动态特性，并指出用户在搜索的同时也在学习，当他们看到检索结果或者其他文档代理时，其信息需求会进行相应的调整。这种动态过程有时称为搜索的*采摘模型*（berry picking model）［157］。

如今的 Web 搜索引擎的快速响应时间，使得搜索用户能够采用一个较为普遍的查询来“试水”，在看到返回结果后，以显示的文字为基础，重构他们的查询方式，试图更“接近”所需的目标［158，755，1082］。例如，一个复杂的查询“一个 1/2 英寸的燃气烧烤炉软管连接器，用于 3/8 英寸的家用插座”，这个查询很可能是失败的，典型的搜索用户会选择一个更为普遍的查询，如“燃气炉软管连接器”，甚至“燃气软管”，查看搜索引擎的返回结果，然后重构查询，或者访问相应的网站，在其中浏览网页，寻找所需要的产品。

这样的做法在采摘方法中是常见的策略，有时也称为*定向*（orienteering）［1219，1569］。进行定向信息搜寻的用户会给出一个快速、不精确的查询，希望近似地得到信息空间的一部分内容，然后再进行一系列的本地导航操作，从而获得更贴近用户兴趣的信息［158，1082］。可用性研究和 Web 搜索日志表明这种方法是常见的。用户很可能会重构他们的查询，一份对搜索日志的分析说明 52%的用户重构了查询［820］。

有些信息搜寻模型关注于搜索过程中使用的*策略*，以及用户在下一个步骤如何做出选择。在某些情况下，这些模型是为了反映专业搜索用户自觉的规划行为。在其他情况下，这些模型是为了捕捉缺少计划性的一般搜索用户的潜在反应。Bates［156］建议，搜索用户的行为可以被搜索策略所刻画，搜索策略反过来由*搜索战术*（tactic）的序列所组成。Bates［156］也讨论了监测当前搜索进度、衡量延续当前策略及改变策略的成本和收益的重要性。

Russell等人［1401］也关注于监测搜索策略的进度，并以成本结构分析或者收益递减分析
23 作为整个过程的目标或是子目标。这种成本结构分析方法，后来被 Pirolli 和 Card 扩展为*信息搜寻理论*（information foraging theory）［1271，1269］，使用进化生物学立场的理论框架，对人们在信息结构内的导航策略进行了建模与预测。

2.2.3 导航与搜索

并非所有的搜索都开始于在搜索框中输入关键词查询。许多网站和一些搜索引擎允许用户通过仔细阅读某种信息结构（information structure）来选择搜索的起点。*导航*（navigation）和*浏览*（browsing）这两个词在这里可以交换使用，它们表示相同的含义——搜索用户通过一系列浏览和选择操作，对信息结构进行查看，并能够在可用信息的多个视图当中进行切换。当信息结构（如在一个网站上的超链接）非常符合用户的信息需求时，用户往往更喜欢浏览而不是关键词搜索。Hearst 等人的研究［737］发现，在多次使用了精心设计的分面分类系统后，自我描述的搜索用户往往会逐渐转变为通过浏览获取信息。

浏览往往是首选，因为*识别*（recognize）出一部分信息要比*召回*（recall）或记住它更为容易。但是，如果花费了过长的时间来寻找感兴趣的标签，或者找不到所需要的信息，那么浏览链接所获得的收益就会递减。也就是说，浏览只有在合适的链接时可用，并对潜在信息具有有意义的提示内容时（有时称为*信息线索*［1269］）才能够有良好的效果。

使用合适的导航结构，某个交互界面可能需要数次点击来引领搜索用户寻找他们的目标，但这并不一定是坏事。Spool［1523］声称，一般来说，搜索用户对于跟踪多个链接并不十分反感，不过他们反感于跟踪那些与他们的目标无关的链接。因此，只要搜索用户在搜寻目标信息的过程中，没有丢失信息的“线索”，交互界面就算表现良好。Spool 讨论了一个用户要寻找某个特定的激光打印机软件驱动程序的例子。假设用户首先点击“打印机”，然后是“激光打印机”，然后按如下的链接顺序：

惠普激光打印机
惠普激光打印机型号 9750
惠普激光打印机型号 9750 的软件
惠普激光打印机型号 9750 的软件的驱动程序
惠普激光打印机型号 9750 在 Win98 操作系统下的软件驱动程序

这样的交互是可以接受的，因为每次细化对于当前的任务都是有道理的，没有一个地方需要后退来尝试另一种选择：即搜索踪迹永远不会变“冷”（即偏离用户需求）。但如果在中途某个时候，搜索用户通过点击没有看到更接近目标的链接，那么这样的经验就会非常令人沮丧，而交互界面从可用性的角度来说就是失败的。

2.2.4 对搜索过程的观察

人们对于搜索过程的研究已经有很多了，获得的成果可以帮助指导搜索界面的设计。这
24 些研究提到的一个共同的观察是用户经常会微调他们的查询，因为这会比第一次就试图给定准确的查询要容易。另一个原因是，搜索用户经常搜索他们先前已访问过的信息［853，1130］，而在看到以前搜索过的材料之后，用户的搜索策略也会相应地有所不同［150，853］。研究人员已经开发出了这样的搜索界面，其中特别考虑了搜索用户重新访问信息的可能性［466，518］，同时支持查询历史和对以前访问过的信息条目进行重新访问。

研究表明，人们难以确定文档是否与主题相关［451，1402，1687］，而人们对一个主题

了解得越少，就越难判断搜索结果是否与主题相关［1516，1620］。对于 Web 搜索引擎，搜索用户往往只关注排名靠前的搜索结果，而偏颇地认为排名第一或第二的文档要好于那些排名较低的文档［663，844］。

研究还表明，人们很难估计在搜索结果中有多少是相关的，他们对一个主题越不了解，也就越有可能自信地认为相关信息都已经访问过了［1551］。此外，人们往往在找到几个结果后就终止搜索过程，即使文档集中可能还会有更好的结果［1547］。

有些搜索可用性的研究评估了搜索过程本身的影响，并对专家和新手进行了对比，虽然这种划分形式还没有达成共识的分类标准［81］。研究指出，专家会使用与新手不同的搜索策略［771，990，1687］，但也许更说明问题的是，其他研究发现了搜索知识和领域经验之间的交互效应［771，832］。在一项研究中，能够找到高质量文档的杰出分析师的总体特点是分析的持续性，那些阅读更多文档、花费更多时间的人比其他人完成得更好［1247］。在另一项研究中，搜索专家比新手更耐心，并有积极的态度，这往往会带来更好的搜索结果［1551］。

2.3 现今的搜索界面

典型搜索会话的核心过程是由查询描述、搜索结果检查和查询重构组成的。随着搜索过程的进行，搜索用户会更加了解他们想要的主题，以及可用的信息来源。

本节将要介绍几种用户界面的组件，它们已经成为了搜索界面中的标准，并表现出了很高的可用性。在描述这些组件的同时，我们也将介绍它们所支持的设计特点。在理想的情况下，这些组件被集成在一起，以支持搜索进程的不同部分，但分开讨论会更有助于我们对它们的了解。

2.3.1 启动搜寻

信息搜寻的过程是如何开始的？在今天，网络已经在很大程度上取代了传统的物理信息源，如电话簿和百科全书等。对于网上信息系统的用户，开始搜索会话的最常用的方法是访问 Web 浏览器，并使用 Web 搜索引擎。 25

另一种开始搜索的方法，是从以前访问过的网站收藏中选择一个网站，这些收藏通常存储在浏览器中的书签中。这种方法曾经被大量地使用，然而随着搜索引擎服务变得更快捷也更准确，这种方法就不再那么流行了［1569］。在其他一些书签系统中，用户将偏爱的网站链接存储在一个网站上（因此从任何连接的计算机都可以访问），其中还可以看到其他人都选择保存了什么网址，这种书签系统已经在一小部分用户中深受欢迎。这些网站（delicious.com 和 furl.net，即现在的 diigo.com，就是这方面的例子）允许用户设定内容的标签（label 或 tag），按主题搜索或浏览，以及按网站标题进行文本搜索。

网站目录曾经也是一个常见的出发点。在较早的时候，Yahoo.com 的目录在当时是最流行的导航起点，但现在网络目录已基本上被搜索引擎所取代，一方面因为网络规模变得太大，没办法手动构造目录，另一方面也因为 Web 搜索的精度不断提高［1267］。不过，有一些学者认为，搜索用户应该对信息的来源有更多的认识，并认为在搜索结果列表中，这些信息应更加突出地显示［1355］。如果想了解更多关于网站目录的信息，请见 11.8.2 节。

2.3.2 查询描述

一旦选定搜索起点，用户表达自己信息需求的主要方法就是在搜索框中输入一些词语或

者从目录以及其他信息组织中选择链接。对于 Web 搜索引擎来说，查询是通过文本形式指定的。如今这通常是通过在键盘上输入文字的方式来实现的，但在未来，伴随着我们逐渐开始以移动设备作为输入媒体，通过语音命令进行查询的方式有可能会越来越普遍。

在如今的 Web 查询中，输入的文字通常很短，一般由 1～3 个词语组成 [820，819]。多词查询往往视为一个短语，但查询也可能是由多个主题所组成的。短查询反映了标准的使用场景，用户查看搜索引擎返回的结果。如果结果是不相关的，用户会重构他们的查询；如果结果是令人满意的，用户就会定位到最相关的网站，在该网站上继续微调查询 [158，539，755，1082]。这种先用普遍的查询来寻找信息空间中有用的部分，然后跟随相关网站超链接的搜索行为，是 Web 搜索当中应用定向策略的一个示范 [1219，1569]。有证据表明，在许多情况下，用户虽然倾向于更详尽地表示他们的信息需求，但过去的搜索引擎使用经验告诉他们，这种方法不能很好地工作，而关键词查询与定向相结合会表现得更好 [201，1288]。

在 Web 搜索出现之前，商业文本搜索系统通常支持布尔运算和基于命令的语法，而实际上并没有支持关键词查询。但是，布尔运算符和命令行语法已经被一再地证实难以让大多数用户理解，试图使用它们的人经常会犯一些错误 [499，672，699，755，763]。

虽然大多数 Web 搜索引擎支持一些布尔形式的语法，但最近一项针对 Web 查询日志的
26 研究表明，在超过 150 万的查询中，仅发现 2.1%含有布尔运算，7.6%含有其他查询语法，主要是双引号短语 [819]。另一项研究考察了近 60 万用户在 2006 年期间，总计时间超过 13 周、数百万的交互日志。他们发现，1.1%的查询包含 4 个主要的 Web 运算符（双引号、＋、－和 site:）中至少一个运算符，只有 8.7%的用户始终使用运算符 [1685]。7.2.1 节将介绍更多关于 Web 查询的内容。

Web 排序已经经历了 3 个主要阶段。第一阶段大约从 1994—2000 年，大多数的搜索引擎使用统计排序，但是没有使用网页内查询项的（位置）邻近信息和网页相对重要性的信息。那时，整个 Web 的规模还比较小，不太可能有相关的信息源为那些较为复杂的查询提供答案。并且有可能会检索出那些缺失查询中关键词的网页，许多用户无法理解这样的行为方式。（例如，AltaVista 引入了强制运算符，用加号表示，即允许用户可以在一个词前增添一个加号，表示这个词必须出现在查询中，但只有那些极富经验的用户才会利用这种查询运算符。）

在 1997 年左右，谷歌转向了只采用合取查询的方式，这意味着只有所有查询项都出现在网页中时，网页才会被检索到。他们还增加了查询项的邻近信息和网页的重要性打分（见 11.5.2 节的 PageRank 算法），这大大提高了许多查询的相关性，特别是导航查询；比如，以“丰田”（Toyota）作为查询，会检索到丰田公司的主页，而不是那些“丰田”出现次数最多的页面。其他的 Web 搜索引擎也紧跟着这种趋势，合取排序成了常态。

随着网络上可用信息数量的增加，老练的搜索用户发现，把较长的查询看做短语往往会找到高度相关的结果。过去，如果搜索用户有复杂的信息需求，并试图充分地表达给 Web 搜索引擎时，这样的尝试往往都会失败。例如，如果一个搜索用户想知道“我在哪里可以找到 1985 年的卡罗拉的轮毂？”以这种形式编写的查询由于合取约束，将无法返回任何结果。现在，Web 搜索引擎已经变得越来越精细，能够去掉一些无意义的项，而只匹配重要的查询项，在排名较高的文档中，这些查询项彼此相邻。另外，可以使用其他在 Web 搜索中已经证明有效的方法进行排序。有关查询语言的更多细节见 7.1 节。

2.3.3 查询描述界面

文本查询的标准界面是一个搜索框，用户输入查询时，通过按键盘上的回车键或点击与

表单相关的按钮进行查询。研究表明，查询长度与输入框宽度之间有一定的关系；小的输入框会阻碍长查询，而宽形式的输入框则会鼓励长查询 [171，585]。

有些输入框被分为多个组件，允许用户更自由地输入查询文本，并跟随着一些查询过滤的输入框。例如，在 yelp. com 上，用户在第一个输入框中输入一个普遍的查询，通过在第二个输入框中输入位置信息，对搜索进行改进（见图 2-1）。表单允许选择以前用到的信息，这些信息有时是结构化的，并允许设置为未来使用的参数。例如，yelp. com 的表单会显示用户的本地位置（如果过去曾经指定过）以及其他近期指定过的位置，并可以选择添加额外的位置。 27

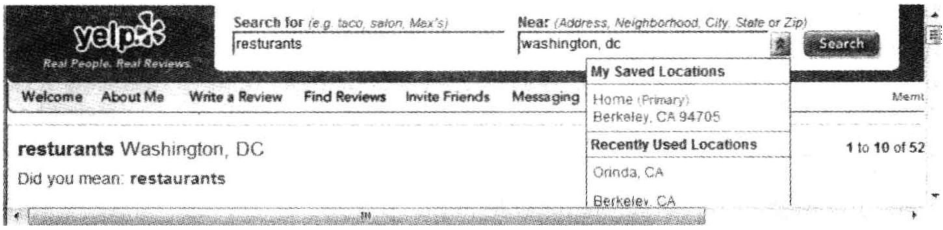


图 2-1　yelp. com 的查询表单，显示了对结构化查询以及存储之前查询信息的支持

一个在搜索框中使用得越来越普遍的策略是，通过灰色文字来暗示什么类型的信息应该输入到搜索框中。例如，在 zvents. com 搜索中（见图 2-2），第一个搜索框上标有“你要买什么?”，而第二个框标有“什么时候（今晚、本周末、……)”。当用户将光标放在搜索输入框上时，灰色的文字消失，用户可以输入自己的查询项。

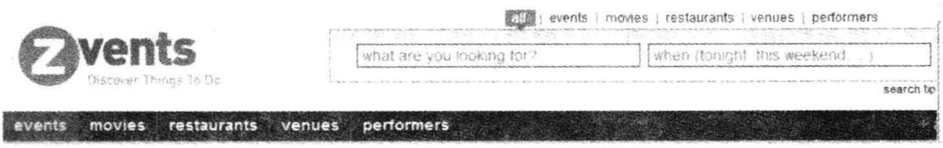


图 2-2　zvents. com 的查询表单，表单中的灰色文字说明什么类型的信息可以被输入

这个例子也说明了现在的搜索引擎支持专用的输入方式。例如，网站 zvents. com 会识别诸如“明天”一类对时间敏感的查询词，并以事先设定好的方式来进行处理。它还能够更加灵活地处理更正式的日期格式，因而搜索“星期三”（wed）的“喜剧”（comedy）时会自动计算最近的星期三的具体日期。这是一个很好的例子，说明我们应该通过设计界面来反映人们是如何思考的，而不是要求用户遵循那些不可靠且流于形式的标准程序来思考。（这种放宽查询描述的方法，更适用于那些“非正式的”（casual）界面，在这些系统当中，日期并不是最关键的要素。非正式的日期格式在填写税表时是不能接受的，因为发生错误的代价太大了。）

一个已经显著改善了查询描述的创新是动态生成的查询建议列表，当用户输入查询时，表单实时显示查询建议 [1684]。这种方法也称为*自动填充*（auto-complete）、*自动建议*（auto-suggest），或*动态查询建议*（dynamic query suggestion）。通过对大规模的日志进行研究发现，用户在大约 1/3 的时间里，点击了雅虎搜索助手提供的动态查询建议 [61]。这一主题将在 11. 7. 2 节介绍 Web 搜索引擎时进行详细解释。 28

通常显示的查询建议是那些前缀字符与之前输入的字符匹配的词语，但在某些情况下，显示的是只有中间字符匹配的词语。如果用户输入多个词的查询，那么显示的查询建议可能是之前输入内容的同义词，但在词法上并不匹配。举例来说，Netflix. com 用灰色字体显示可能需要的词，然后通过一个下拉列表框显示可以点击的词语。

在动态查询建议界面中，匹配的显示也有着不同的方式。有些界面根据类别信息对建议进行着色。在大多数情况下，用户必须移动鼠标到所需的查询建议上以选择它并用来填充查询框。在某些情况下，查询可以立即进行；在另一些情况下，用户必须输入回车键或点击"搜索"按钮才能进行查询。

查询建议可能来自多种资源。在某些情况下，列表是根据用户自己的查询历史获得的，在其他情况下，它基于其他用户的热门查询。这个列表也可以来自于网站设计人员认为重要的一组元数据，例如在药理文献搜索时显示的一组已知疾病或基因的名字（见图2-3），在电子商务网站搜索时显示的产品列表，或者在电影网站上搜索时显示的热门电影列表。这些建议也可以来自网站内部的所有文本。

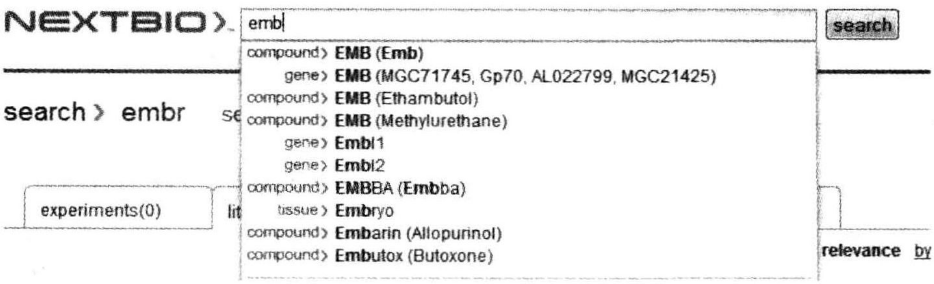


图2-3 按类型分组的动态查询建议，源自NextBio.com

查询描述的另一个形式包括从一些信息显示中进行选择，最典型的是在超链接或保存的书签中选择。在某些情况下，选择一个链接，除了结果列表，还会产生更多的链接来进行下一步导航。这种查询描述会在2.3.6节中进行详细的讨论。

2.3.4 检索结果显示

当显示搜索结果时，或者是显示全部文档，或者将文档的一些有代表性的内容提交给搜索用户。这种文档代理（surrogate）指的是文档的摘要，这是一个成功的搜索界面的重要组成部分。而文档代理的设计和检索结果显示是目前研究和实验比较活跃的领域。

文档代理的质量极大地影响对搜索结果列表相关性的感知。在Web搜索中，页面标题
29 通常与URL一起加亮显示，有时也会与其他元数据一起显示。在对信息集合进行搜索时，出版日期和作者等元数据往往会显示（但这类元数据较少应用于网页）。文本*摘要*（summary）（也称为*摘要*（abstract）、*提取*（extract）、*摘录*（excerpt），或*片段*（snippet））包含了从文档中提取的文本，它们对检索结果的评估是至关重要的。

一项研究评价了搜索结果中的哪个属性会获得更多的点击，并从中找到了许多能带来正面效果的因素，包括更长的文本摘要、包含查询关键词的标题、标题组合、包含作为短语匹配的查询的摘要和网址（URL）、更短的URL，以及域名中包含查询项的URL[390]。

目前，标准的结果显示是一个文本摘要的垂直列表，有时也称为*搜索引擎结果页*（Search Engine Results Page，SERP）。在某些情况下，摘要是对包含查询项的文档的摘录。在其他情况下，通过*混合*（blended）结果（也称为*全能搜索*，universal search）技术，有些特殊的元数据与标准的文本结果一起显示给用户。例如，以"彩虹"作为查询，返回的搜索结果可能包含一行彩虹的示例图像（见图2-4），或者查询运动队的名称可能检索出最近的比赛得分和一个购买门票或浏览比赛直播时间表的链接（见图2-5）。Nielsen[1206]指出，

在某些情况下，搜索结果列表可以直接满足信息需求，从而使搜索引擎变成“答案引擎”。



图 2-4 雅虎搜索中对于查询“彩虹”的搜索结果页面。结果从上到下分别包括：查询改善建议，彩虹图片的链接，关于彩虹的百科文章和一些说明的图片，以及名为“彩虹”的摇滚乐团的百科文章



图 2-5 谷歌搜索引擎对于查询“火箭”的搜索结果界面。根据查询词的不同意义，显示了不同种类的信息。第一个是篮球队主页的链接，以及一些该网站内部的“深度链接”(deep links)。接下来，是与球队有关的其他链接，关于火箭的百科页面，以及火箭队的视频链接和如何制造火箭的链接

对排序的研究发现，邻近性信息可以很有效地提高搜索的精度 [391，733，1563]。可
30 用性研究建议，将查询项出现在文档时的上下文显示出来，可以有助于用户评估结果的相关性 [1593，1682]。这有时称为上下文关键字（Keywords in Context，KWIC）、查询偏置摘要（query-biased summary）、面向查询的摘要（query-oriented summary），或用户主导摘要（user-directed summary）。

查询项的加亮显示（highlighting）可以在视觉上改善搜索结果列表的可用性，这个观点已经有几十年了 [974，1005，1063，1082]。加亮是指在视觉把某个部分和其他部分进行区别，可以通过粗体文字、改变文字或背景的颜色、改变文字的大小，以及其他方法来实现。加亮显示既可以用于搜索结果中的文档代理，也可以用于检索出的文档本身。有些界面通过可视化的方法给文档中加亮显示的部分做一个概述 [160，305，661，766]。

确定哪些文字可以用做摘要和多少文字应当显示，是一个具有挑战性的问题。通常，最相关的文章是那些包含所有查询项，并且查询项是彼此紧密相连的，但对于不那么匹配的结
31 果，在显示连续的句子以增加结果的连贯性与显示包含查询项的句子中，要进行一个权衡与取舍。有些研究结果表明，完整地显示整个句子比将句子切分开有更好的效果 [80，1380，1683]，但在另一方面，很长的句子通常也不是我们想要的结果。

还有证据表明，在搜索结果摘要中显示的这类信息应根据查询意图和搜索会话目标的不同而相应地变化。有些研究显示，在某些信息需求下较长的答案会比较短的答案表现得更好 [861，1034，1237]。而当搜索用户决定直接进入到一个知名网站的主页时，简短的结果列表会比长的详细信息更好。在一般情况下，用户对已知项进行搜索时往往倾向于可以指示所需信息的较短的代理。主页搜索本质上是对地址的搜索；用户知道网站的名字，希望找到它的网址（URL）。同样，能够简要说明的事实性信息需求可以被简短的结果满足。相反地，如果用户有一个复杂的信息需求，更深层次的文档摘要可以带来更好的搜索体验。这一点对于那些更丰富的任务来说也是正确的，如建议搜寻或获取相似的主题。

其他种类的文档信息可以有效地显示在搜索结果页面中。图 2-5 和图 2-7 显示了站内链接（sitelink）和深度链接（deep link）的应用，它们在网站主页的下方显示了网站内部较受欢迎的网页。在另一个例子中，生物科学文献检索研究发现，大多数参与者强烈主张在搜索结果的旁边显示从期刊文章中提取的图片 [736]。在图 2-6 中，对 BioText 系统的截图显示了这种思想，也说明了加亮或粗体显示查询项的作用，以及用户可以或多或少看到查询项的上下文环境的机制。

2.3.5 查询重构

在指定了查询和产生了结果之后，有一些工具可以用来帮助用户重构他们的查询，或将信息搜寻过程引领到一个新的方向。对搜索引擎日志的分析表明，查询重构是一种常见的活动；一项研究发现，在一次会话期间，超过 50% 的搜索用户至少进行了一次查询修改，有接近 1/3 的人进行了 3 次或更多次查询修改 [820]。

在最重要的查询重构技术中，有一种是显示与查询或检出的文档相关的索引项。其中的一个特殊情况是拼写校对或建议；据估计，10%～15% 的查询会出现错字 [461]。在 Web 搜索出现之前，拼写建议主要是基于字典的 [944]。在 Web 搜索出现后，查询日志已应用于开发检测和纠正拼写错误的高精度算法中 [461，1018]。在搜索界面中，通常只有一个更改建议显示；点击更改建议就可以重新执行查询。多年以前，搜索结果中会显示那些据推测不正确的拼写，今天一些搜索引擎已经可以交错显示原始查询的结果和拼写校对后的结果，或将原始查询结果与拼写校对后的结果分别显示。

除了拼写建议外，搜索界面越来越多地采用相关项建议技术，通常称为查询项扩展(term expansion)。对日志所进行的研究发现，如果能提出较好的查询建议，那么在 Web 搜索中它会是一个频繁使用的功能。对日志的研究发现，大约 8%的查询都是由查询项建议产 32
生的 [819]（但它没有显示有多少比例的查询会显示这样的建议），而另一个发现是大约 6%的用户选择点击查询项建议 [61]。

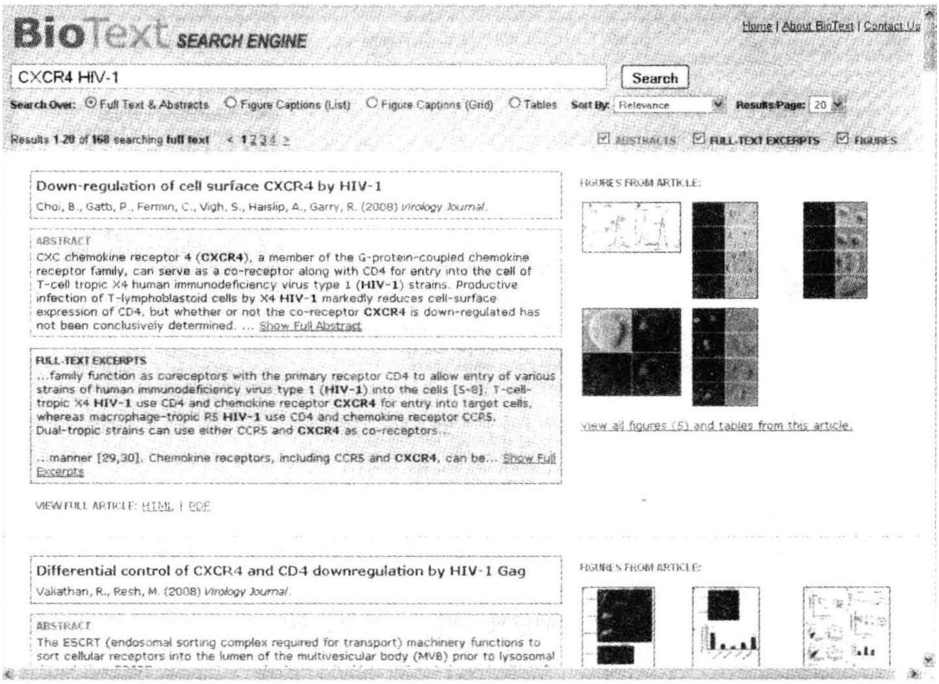


图 2-6 BioText 系统的搜索结果，其中显示了丰富的文档代理信息，包括文章中所抽取的图片、加亮或黑体显示的查询项，以及扩展或缩短文档摘要的选项，来自 http://biosearch.berkeley.edu

较早使用查询项扩展或建议的工作建立在十几个同义词库上，通常是在显示搜索结果前强迫用户从中选择 [285，492]。最近的研究表明，提供较少的建议，只需要用户进行一次点击，或将相关的查询项组合起来进行一次点击选择，是一种更为可取的方法 [59，501，1681]。图 2-4 和图 2-7 显示了只需一次点击的查询项扩展的例子。

有些查询项建议是基于某一特定用户的整个搜索会话，然而有的则是基于之前提出相同或相似查询的其他用户的行为。一种策略是显示其他用户提供的类似查询；另一种方法是从之前提出相同查询的用户所点击的文档中提取查询项。在某些情况下，相同的算法被用做实时查询自动建议。

相关反馈是另一种方法，其目标是帮助我们进行查询重构，将在第 5 章详细讨论它。其主要思想是让用户指出，对于查询哪些文档是相关的（也可以是不相关的）。在另一些搜索 33
系统中，也可能让用户指出从文章中抽取的哪些索引项是相关的 [918]。系统通过这个信息，可以计算出一个新的查询，并使用某种算法，显示一个新的检索集合 [1402]。

相关反馈已被证明在非交互式或人工设置情况下，都可以大大改进排名顺序 [31，895]。然而，这种方法从可用性角度并不认为是成功的，也没有出现在标准的用户界面中 [451，1402]。这源于几个因素：用户不善于评价特定文档的相关性，特别是对他们不熟悉的主题 [1620，1687]；另外，相关反馈的益处是不一致的，这在可用性方面是有问题的；此外，相关反馈的优势大多体现在需要大量相关文档的任务中，但这在 Web 搜索中并不常

见；事实上有一些证据表明，相关反馈的优势在搜索整个 Web 时就会消失了 [1570]（大多数相关反馈的优势只是在应用于小规模文档集时才可以显现出来）。

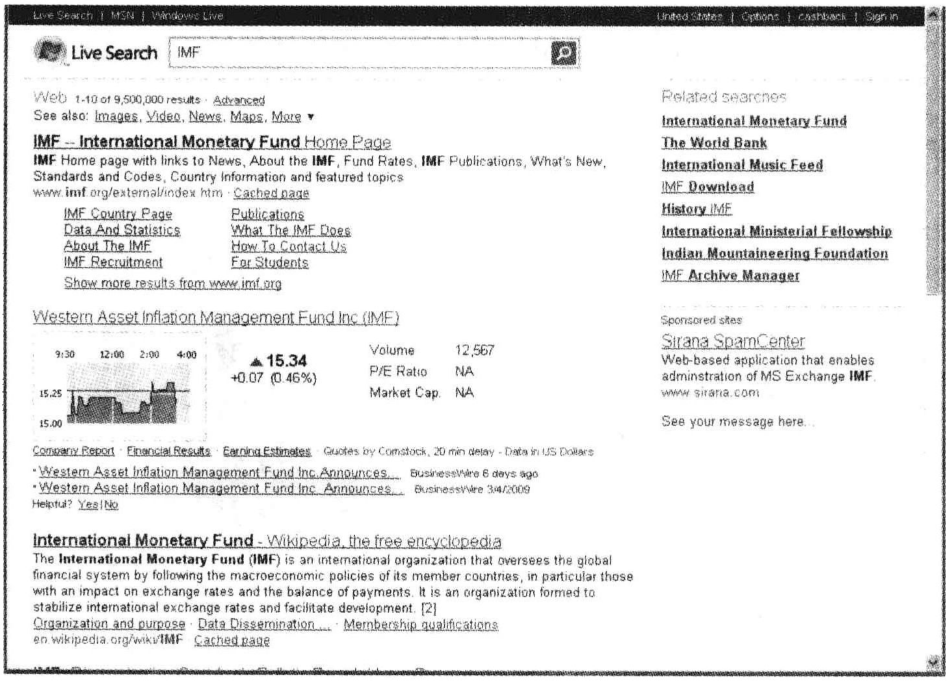


图 2-7 微软的 Live 搜索引擎对于查询"IMF"的结果页面，包括相关的查询建议（在右侧），可选择的"垂直"搜索的链接（图片、视频、新闻等），站内链接如金融统计和一些百科文章

关于相关反馈的变体（自动计算"相关文档"功能）已证明在某些情况下有着积极的影响。在生物医学文献检索系统 PubMed 中，在给定期刊文章的旁边显示一部分相关的文章，
34 这项功能在生物学家中是广受欢迎的。一项研究表明，在显示相关文档的会话中，18.5%的情况下，用户会点击所建议的文档 [1033]。

2.3.6 组织搜索结果

搜索人员通常表示他们希望将搜索结果分成若干有意义的组，以方便理解搜索结果和决定下一步如何做。在一项提取搜索结果分组的纵向研究表明，在应用分组机制的情况下，用户的搜索习惯发生了改变 [862]。现在，有两种搜索结果分组的常用方法：*分类系统*（category system），特别是*分面分类*（faceted category）和*聚类*（clustering）。在本节中，对这两种方法进行详细介绍，对它们的可用性进行比较。

分类系统将一组有意义的标签组织在一起来反映某个领域的相关概念。它们通常是手动构造的，尽管为文档自动设定类别已经可以达到一定的准确率了 [1446]（见第 8 章）。好的分类系统有连贯和（相对）完整的特点，它们的结构也是可预测的，在同一个信息集合内的搜索结果是一致的。

在用于组织搜索结果和表达信息集合结构的分类结构中，最常见的是*扁平的*（flat）、*层次的*（hierarchical）和*分面的*（faceted）分类结构。扁平分类是话题或对象的一个列表。它们可以用做分组、过滤（缩小），或者对搜索界面中的文档集进行排序。大多数网站将信息分类组织，选择相应的类别可以缩小显示的信息集合。在一些实验中，Web 搜索引擎自动

地按扁平分类组织信息；研究表明用户对这种设计给予了积极的回应 [519，945]。在庞大的 Web 内容中找出适用的类别子集是非常困难的，相应地，分类系统对于内容更为集中的信息集合似乎有更好的效果。

在线层次组织（hierarchical organization）在桌面文件系统浏览器中是最常见的。在早期的 Web 中，像雅虎所使用的那种层次化目录系统能够将流行的网站组织成可浏览的结构。然而，当信息的集合变得很大，而且结构之间存在相互的链接时，保持严格的层次化结构就会变得很困难。另外，Web 的大小远远超过了在这个系统中可管理的浏览内容，而搜索引擎的应用极大地替代了目录结构的浏览。

层次化对于目录形式的结果会非常有效，如一本书或较小的文档集。Superbook 系统 [527，528，974] 是一个早期的搜索界面，它用大规模文档的结构来显示查询项命中的情况。在用户指定对一本书的查询后，搜索结果会显示在层次目录中（见图 2-8）。当用户从目录视图中选择一页时，页面会自动显示在右侧，此页内的查询项会被加亮反转显示。最近，有些科研项目应用这个思想来组织企业网 [363，1173，1711]。

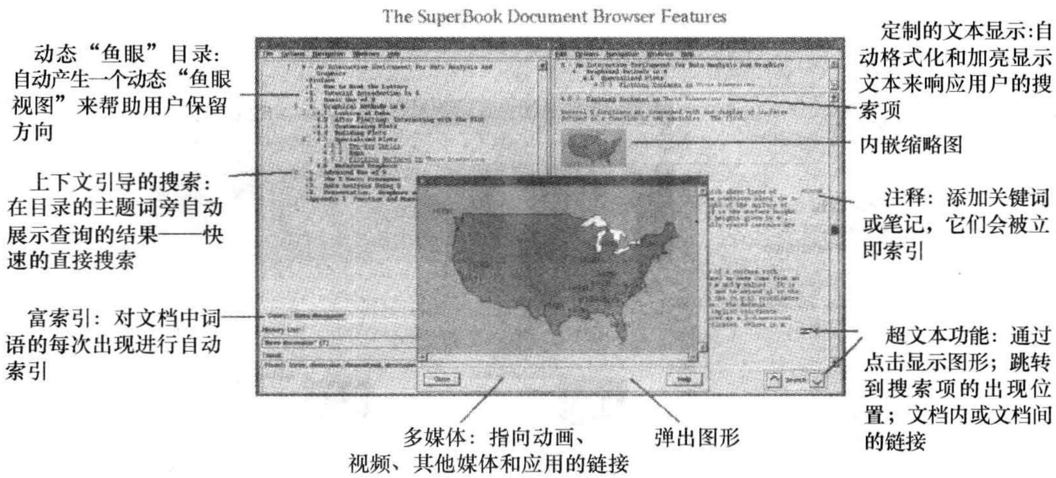


图 2-8　SuperBook 的界面，通过应用一个从大规模手册中制定的目录，在上下文中显示了检索结果 [974]

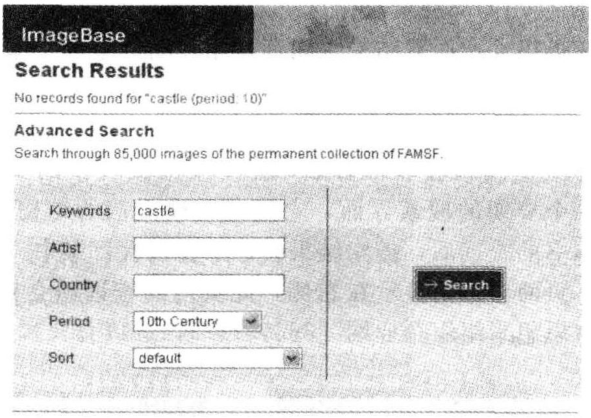
越来越多的人认识到，严格的层次化分类组织对于信息结果的导航并不是理想的选择。层次结构强迫用户从一个特定类别开始，而大多数信息项可能会有许多不同类别的属性。层次化还经 35
常假设信息项仅仅被放置在分类系统的一个地方，然而，计算机界面要比图书馆书架更为灵活。

一种叫做分面元数据的表示方法，已经成为组织网站内容和搜索结果的主要方式。分面元数据是扁平分类和完全知识表示在复杂度上的折中，如果设计得合理，就很容易被用户所理解，相比于其他组织形式也更受青睐。不同于建立大规模的分类层次结构，分面元数据由分类集合所组成（扁平的或层次的），每一个都对应于和需要导航的文档集相关的不同分面（维度或特征类别）。在设计好分面之后，文档集中的每一项都被赋予分面中的若干个标签。

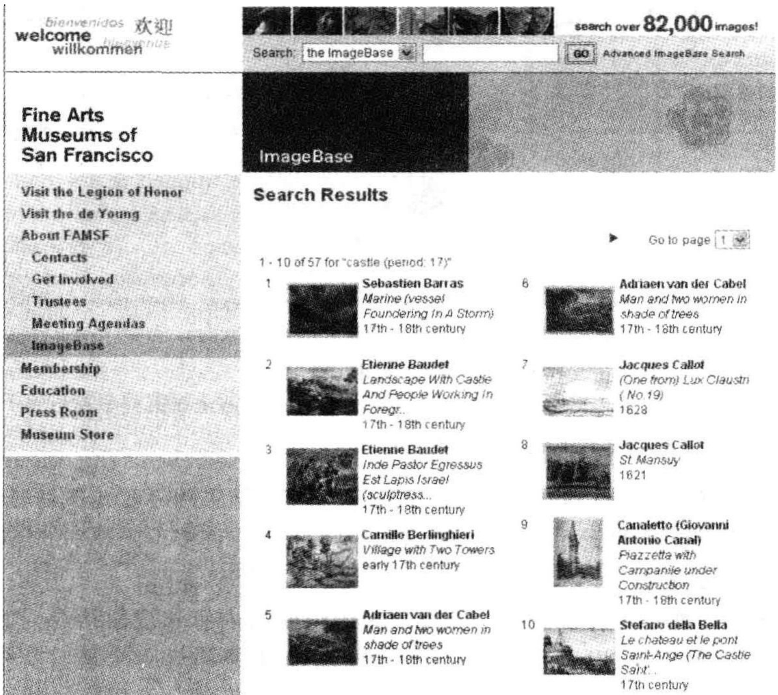
应用了层次化分面导航的界面会同时显示接下来要去的网页的预览和如何在浏览中返回前一个状态，同时将类别结构中的文本搜索无缝地结合进来。从而，用户所需的思考就减少了，因为提高了识别的召回率，同时又保证用户在每次操作时都给出逻辑合理但不常见的选择，最终还保证不会有空的结果集。这种方法提供了组织搜索结果和随后查询内容的一种方案，它可以作为探索和发现过程的重要结构。

图 2-9a、b 显示了在一个假想的搜索会话中，一个典型图片搜索界面的搜索结果。用户

在“Advanced Search”（高级搜索）框中键入关键词“castle”（城堡），并试图选择“10th
36 Century”（10 世纪），但系统返回了错误信息，表示没有记录被找到。在一些试验和错误后，用户发现对于“17th Century”（17 世纪）的搜索可以返回结果，并且该结果以固定的顺序显示，不允许组织和探索。



a）



b）

图 2-9 在旧金山美术馆图片集上的典型图片搜索界面。a）用户在典型的“Advanced Search”（高级搜索）框中的两个字段输入查询后出现的错误信息。这种输入框的常见问题是产生空的答案集。b）通过“Advanced Search”（高级搜索）框搜索关键词“castle”和时间区间“17th Century”的标准搜索结果列表

图 2-10 显示通过应用分面导航的 Flamenco 系统，同种类的信息可以获得更好的可理解性[737，1746]。用户最初键入查询关键词“castle”（城堡），搜索结果显示了 229 个图片，左侧结果可以允许用户通过 Media（媒体）类型、Location（地点）、Object（对象）（图片内可见）、Building（建筑物）类型（“castle”（城堡）是其中一种），或 Author（作者）等信息来组织答

案的结构。由于用户能够选择超链接，并且查询预览［1281］可以显示在链接被点击后可以看到多少结果，因此空的答案列表不再是一个严重的问题。在这个例子中，用户首先选择“Media>Prints”（媒体>印刷品），然后再由“Location>Europe”（地点>欧洲）对结果进行组织，然后通过选择“Print”（印刷品）下面的分支重新进行组织。左侧的层次化分面元数据显示了剩下的 197 张图片属于哪个欧洲国家，以及出现的次数。选择一张图片会显示与其相关的元数据，并附有相关概念的链接，如“ruins”（废墟）和“hill”（小山）。图 2-11 显示了如何将类似的想法应用于数字图书馆目录，图 2-12 显示了它应用于黄页和网站预览的情形。

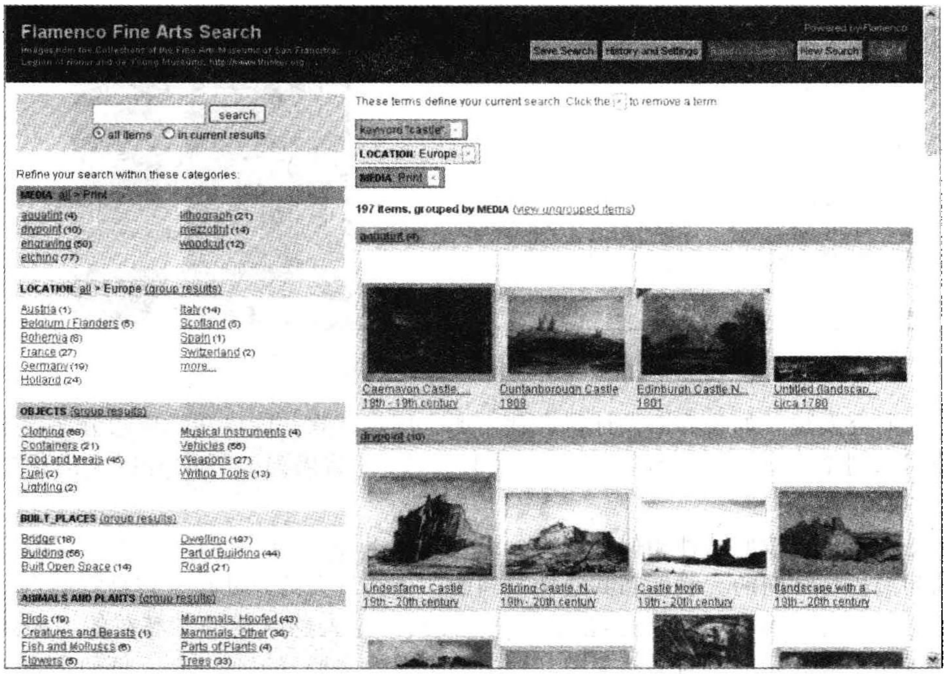


图 2-10　Flamenco 界面的分面导航，应用在旧金山美术馆图片集的一个子集上　37

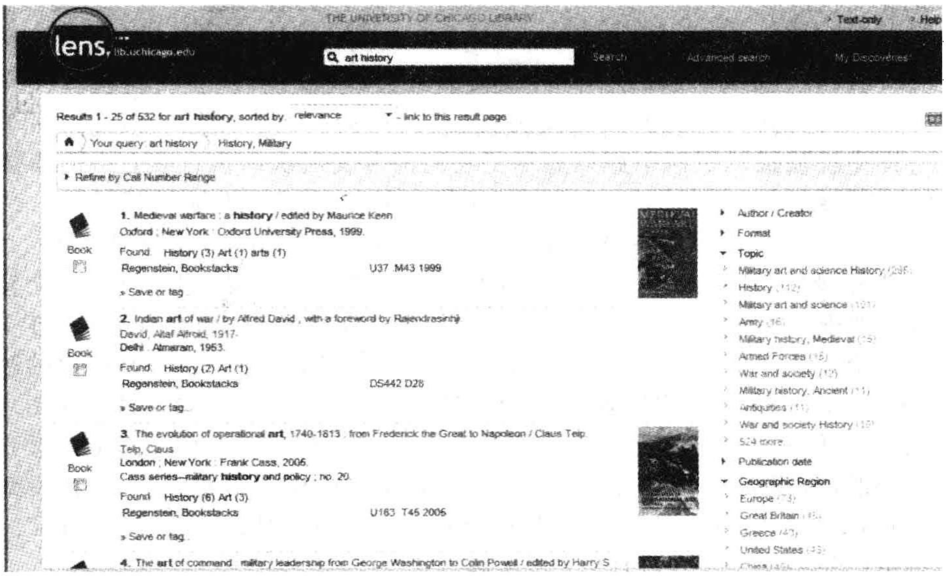


图 2-11　芝加哥大学数字图书馆的分面导航，来自 AquaBrower



图 2-12 yelp. com 的分面导航

可用性研究发现，如果界面设计得合理，那么用户喜欢并能成功地应用分面导航。在用做集合搜索和浏览时，分面界面相对于标准的关键词-结果列表界面有巨大的优势［737，
38 1746］。

聚类是指将条目按照某种相似度进行分组（见第 8 章）。在文档聚类中，相似度一般通过计算词和短语特征间的关联性和共通性得到的。聚类最大的好处在于它是完全自动的，很容易应用于任何的文本集。聚类还能反映一组文档中令人感兴趣的、潜在的、未知的新趋势，并能将那些彼此相似但与文档集中其他文档不同的文档分组到一起，例如出现在主要语种为英语的文档集中所有用日语写的文档。

聚类的缺点包括形式和结果质量的不可预测性、标记分组的难度，以及聚类层次化的反直觉性。有些算法［862，1764］在占主导作用的短语间建立簇（cluster），来构造可理解的标签（见图 2-13），但是每个簇的内容不一定是彼此连贯的。

图 2-14 显示了 Vivisimo 的 Clusty 系统在查询“senate”（参议院）时，搜索引擎结果的聚类输出。图中扩展显示了两个簇以表示了它们的分支层次结构。最上层的簇被标记为“Biography，Constituent Services”（传记，选举服务），其子簇分别被标记为：“Photos”（图片）、“Issues/news”（出版物/新闻）、“Visiting Washington”（访问华盛顿）、“Voting record”（投票记录）、“Virginia”（弗吉尼亚州）和“Maine”（缅因州）等。每个簇代表什么并不是非常明确；如果它代表的是美国参议院，那么在其他簇中也会有很多关于美国参议院的页面。无论如何，最上层的标签并不能代表具体的信息。下一个最高层次的簇标签为“Senate Committee”（参议院委员会），选择它后则会显示相应的组成文件（在图片右侧），
39 从美国参议院主页（关注的不是其下属的多个委员会）到某些具体的美国参议院委员会的网页，再到堪萨斯州和柬埔寨的页面。第三个主要的簇“Votes”（投票），也扩展到一些如“Constituent Services”（选举服务）、“Obama Budget”（奥巴马预算）、“Expand”（扩张），以及“Senate Calendar”（参议院日历）的子簇。

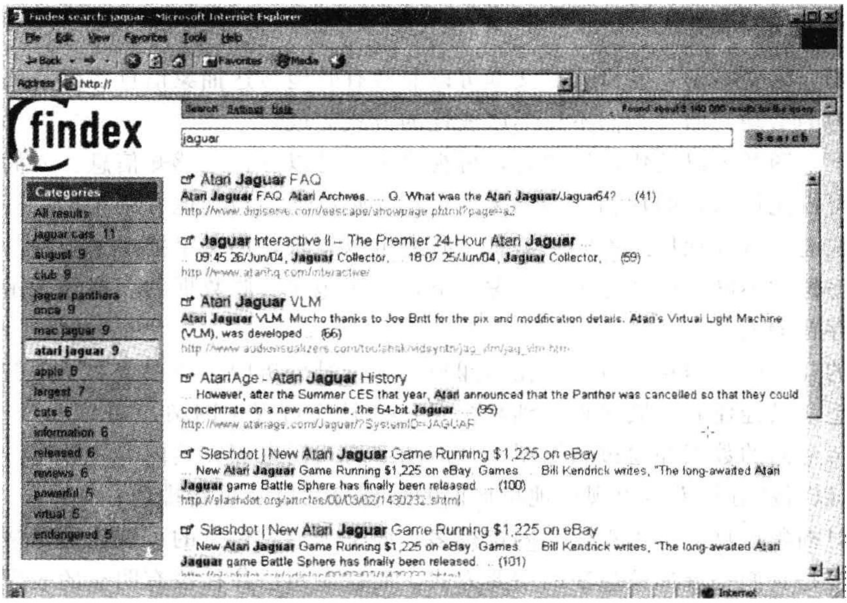


图 2-13　应用 Findex 聚类 [862] 产生的输出，来自 FindEx. com Inc. © 2010 和许可方

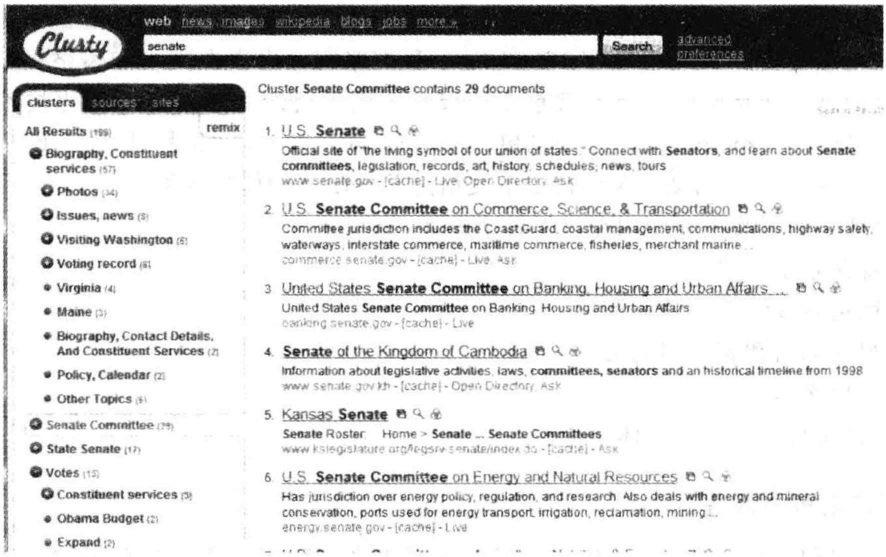


图 2-14　查询“senate”的聚类输出，来自 Clusty. com

话题的混合性和分组之间的重叠，对文档聚类是很典型的。可用性结果表明，用户不喜欢聚类产生的无规律的分组，而是更喜欢可理解的、并通过统一的层次颗粒度来表示的层次化结构 [1301，1376]。

分面界面相对于聚类的一个缺陷是感兴趣的类别必须是预先知道的，所以数据中一些重要的新趋势可能不会注意到。尽管建立分面层次化结构的尝试正在不断推进之中 [1536]，但到目前为止，最大的缺陷是在大多数情况下，类别的层次化结构是手动建立的，而自动建立类别只取得了部分的成功。

2.4 搜索界面的可视化

本书主要介绍文本信息的检索。文本可以非常有效地传达抽象信息，但是阅读甚至浏览文本都是费力的活动，而且人们不得不以线性的方式完成。

40~41 相比之下，图像可以被快速地浏览，可视化系统可以并行地感知信息。人们可以很好地理解图像和可视化信息，图片和图形更迷人也更有感染力。与其他方法相比，信息的可视化表示可以更快速和有效地传达不同的信息。我们可以想象，用文字描述一张脸与显示一张脸的图片会有怎样的不同，或者还可以考虑一下，一张包含关联数据的表格与表示相同信息的散点图之间存在的差异。

在过去的几年中，信息的可视化在新闻报道和金融分析中已经很普及了，关于信息可视化的创新性想法已经蓬勃发展并扩展到整个 Web。社交可视化网站，如 ManyEyes [1637]，允许用户上传他们的数据并通过条形图、气泡图或折线图来探究其内容，数据分析工具，如 Tableau，可以帮助分析人员可视化地将他们的数据分片以及重新排列。

然而，对抽象信息进行可视化要困难得多，而文本形式信息的可视化更是格外有挑战性的任务。语言是我们交流抽象想法的主要方式，而这些想法往往没有明显的表现形式。词语和概念没有内在的顺序，这使得词语很难通过坐标来画出有意义的图。

尽管有这些困难，但研究人员还是试图通过信息可视化技术来表示信息获取过程的各个方面。除了应用图标和颜色的加亮显示之外，他们还常常使用线条、圆圈和画布形式的空间布局来作为信息视图。Sparklines [1606] 是一种缩略图，内嵌在文本和表格里显示。对于可视化抽象信息，交互性似乎是非常重要的属性；最主要的交互信息可视化技术包括平移 (panning) 和缩放 (zooming)、变形视图 (distortion-based) (包括焦点加上下文 (focus plus context))，以及使用动画来保持上下文信息并使闭塞的信息可见。

搜索可视化的实验主要应用在以下各个方面：

- 可视化布尔语法
- 可视化查询结果中的查询项
- 可视化词语和文档间的关系
- 可视化文本挖掘

接下来将对每一项进行具体讨论。

2.4.1 可视化布尔语法

正如上文所提到的那样，布尔查询的语法对于大多数用户来说是有困难的，也很少应用于 Web 搜索中。很多年来，研究人员已经实验了如何通过可视化布尔查询来使查询更容易地为人所理解。一个常用的方法是可视化显示韦恩图 (Venn diagram)；Hertzum 和 Frokjaer [755] 发现简单的韦恩图表示可以获得比布尔语法更为准确的结果。这种想法的一个更为灵活的版本可以在 VQuery 系统上看到 [851] (见图 2-15)。每一个查询项用一个圆圈或椭圆表示，圆圈间的交集代表查询项之间的 AND 运算 (逻辑合取)。VQuery 通过画布活动区域内的圆圈集合表示逻辑析取，通过活动区域内对圆圈的取消选定来表示逻辑非。

布尔查询的一个问题是，它们很容易最终产生空结果或者太多的结果。为了纠正这个问

42 题，过滤流可视化允许用户为查询设计不同的分量，然后通过图形流的方式显示在应用每个

运算符后有多少搜索结果 [1755]。布尔查询的其他可视化表示包括垂直和水平的排列区块 [60]，以及将查询项的分量表示为重叠的“魔术”镜头 [566]。

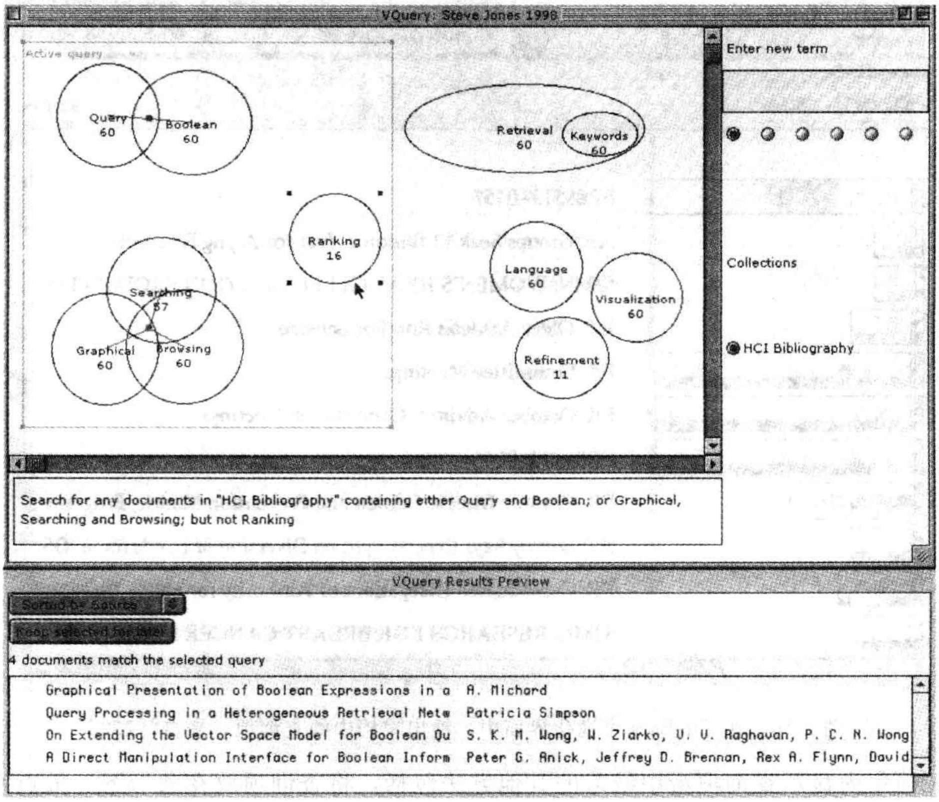


图 2-15　VQuery [851] 为布尔查询制定的韦恩图界面

2.4.2　可视化查询结果中的查询项

如上文所讨论的那样，理解查询项在检出文档中所扮演的角色有助于对相关性的评估。在标准的搜索结果列表中，通常会选择那些包含查询项的句子作为摘要，而这些查询项出现在标题、摘要和网址时会被加亮或黑体显示。从可用性方面看，这种加亮显示的方法已经证明是有效的。

人们已经设计出了许多实验性的可视化界面来明确这种关系。最有名的是 TileBars 界面 [732]，其中文档用水平布局图（horizontal glyph）显示出来，命中的查询项在布局图中的相应位置标出（见图 2-16）。它鼓励用户将查询项拆分成不同的分面，每一行有一个概念， 43
每一个文档表示内的水平行说明了每一个主题下查询项出现的频度。较长的文档被分为子话题的片段，其方法或者通过段落或章节分割标记，或者通过一项叫做 TextTiling [731] 的自动段落划分技术。颜色的灰度表示了查询项出现的频度。可视化显示表明了不同的查询主题在文档中重叠的情况。

人们还设计出了一些其他的 TileBars 显示形式，例如为每个查询项显示一个正方形的简化版，在这个版本中，使用了颜色分层来表示查询项的频度 [770]。两个更为精简的版本在文档图形的按钮中用灰色显示了查询的命中结果 [741]，或者在饼图中用彩色显示命中结果 [51]，但这些视图并不能显示查询项的位置重叠情况。

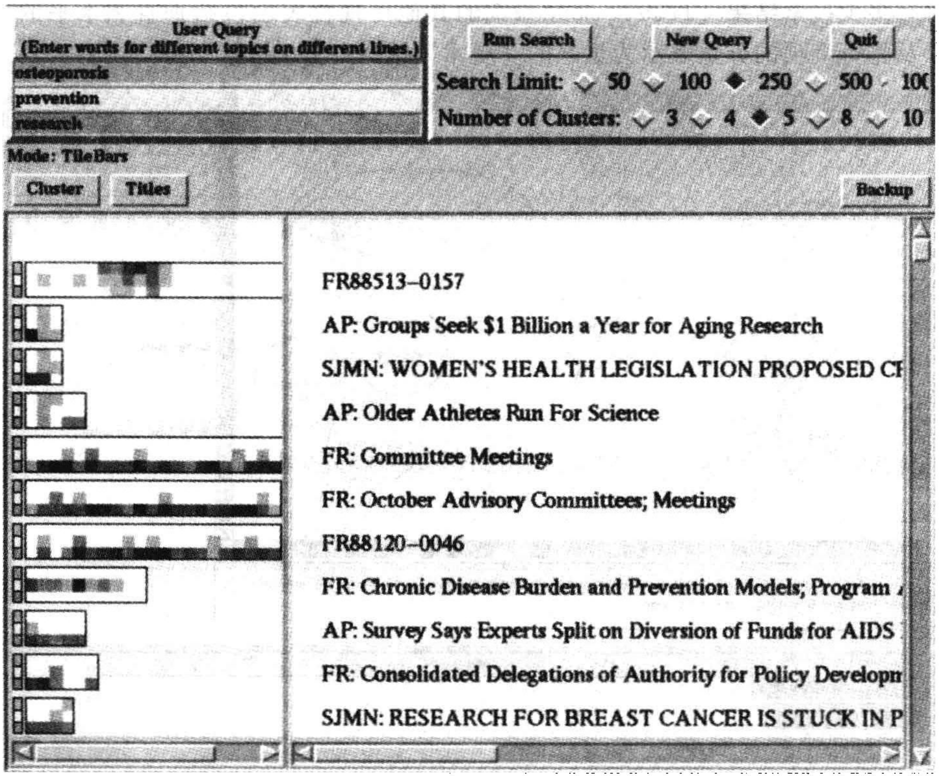


图 2-16 在 TileBars 可视化界面中，检出文档中的查询项，来自 [732]

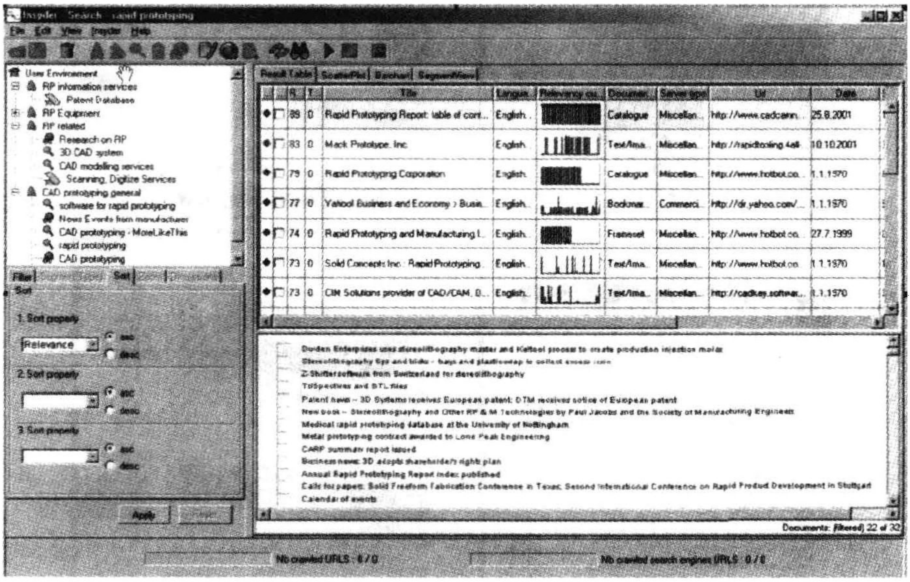
在文档集内显示查询项命中结果的其他方法包括，将查询项放在条形图、散点图和表格中。Reiterer 等人 [1342] 的可用性研究比较了五种视图：标准的 Web 搜索引擎形式的结果列表视图；包含了标题和文档元数据，用图形显示在文档内部的查询项命中位置，并通过
44~45 高度来表示频率的列表视图（见图 2-17）；彩色的 TileBars 类型的视图，其中的文档标题显示在图像旁边；像 Veerasamy 和 Belkin [1630] 那样的彩色柱状图视图；表示相关性得分及其发布日期的散点图视图。

在被问及主观感受时，40 个参与者大体上都会首先选择字段可排序的视图，然后是 TileBars，最后是网页样式的列表。柱状图和散点图则有很多负面反应。对于任务有效性而言，其他方法和网页样式的列表没有明显的不同（除了柱状图之外，其效果会差很多）。所有其他方法都比网页样式列表花费更多的任务时间。最后一点说明了可视化搜索中一个常见的结果——即使在可视化被认为有助于任务的情况下，它通常也比只有文本的界面花费更长的搜索时间。这可能是因为从解释图像转换到阅读文本需要花费一些时间，因为它们属于不同的认知功能。

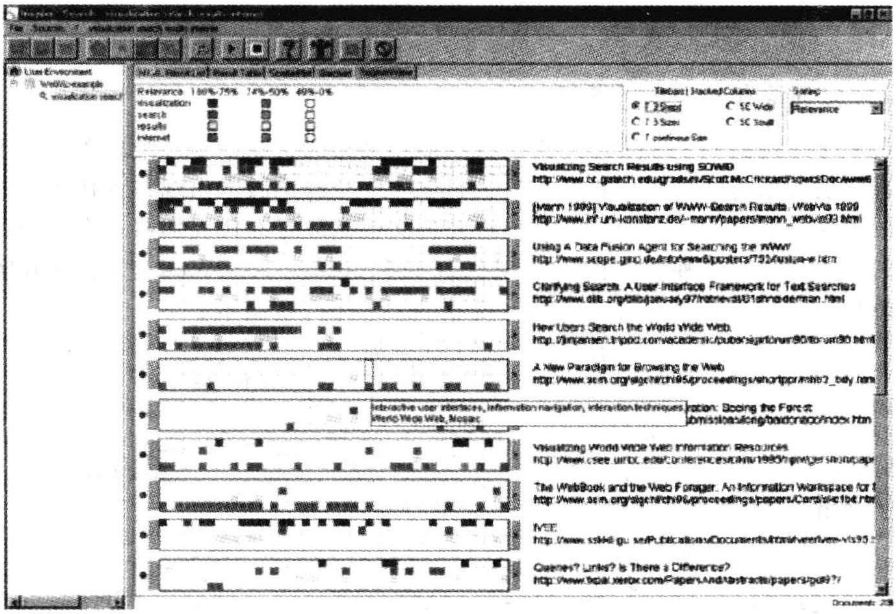
另一种在文档内部显示命中查询项的想法是显示缩略图——文档视觉外观的缩略版（见图 2-18）。一项应用缩略图的实验发现它们在改进搜索结果方面不会比空白区域更好 [468]，另一项实验发现参与者更容易错误地认为显示了缩略图和标题的文档是相关的
46 （相比于那些只显示缩略图的情况）[523]。这两项研究都显示了缩略图对用户有主观上的作用。

负面的研究结果可能源于缩略图大小的问题，更新的结果表明增加缩略图的大小可以提高搜索结果的可用性 [858]。一项相关的研究表明通过加亮显示缩略图内部的查询项，使其

更大并且更可见，对于某些种类的任务来说可以提高搜索结果的可用性［1720］（见图 2-18）。在 SearchMe 搜索引擎中，已经开始在搜索结果中使用更大的缩略图和相应增大的文本，而这是通过封面流（Cover-flow[⊖]）动画来表示的。



a）



b）

图 2-17　a）字段可排序的搜索结果视图，包括一个柱状图风格的图形界面，表示了查询项命中的位置，这是一个简化版本的 TileBars，来自［M. A. Hearst，*Search User Interfaces*，Cambridge University Press，2009，figure 10.17a］［735］；b）彩色版的 TileBars 视图，来自［1342］

⊖ 即由苹果公司首创的将多首歌曲的封面以 3D 界面形式显示出来。——译者注



图 2-18 带文本增强功能的缩略图，来自 [1720]

2.4.3 可视化词语和文档间的关系

很多可视化系统的开发人员已经提出将词语和文档放置在一个二维画布上的想法，其中符号的邻近代表词条与文档语义相关。这个想法的一个较早版本出现在 VIBE 的界面中，其中查询项放置在一个平面上，包含查询项组合的文档被放置在代表那些查询项的图标的中间（见图 2-19）[1228]。这种想法的一个现代版本在 Aduna Autofocus 产品中出现，另外在 VIBE 的基础上，Lyberworld 项目 [744] 制作出了一个 3D 版本。

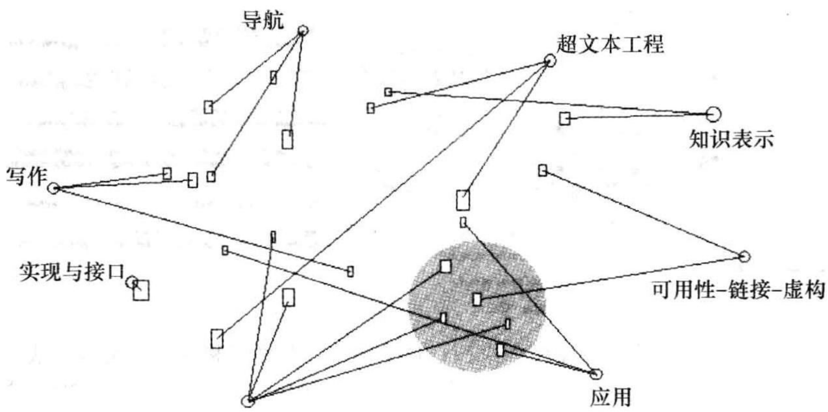
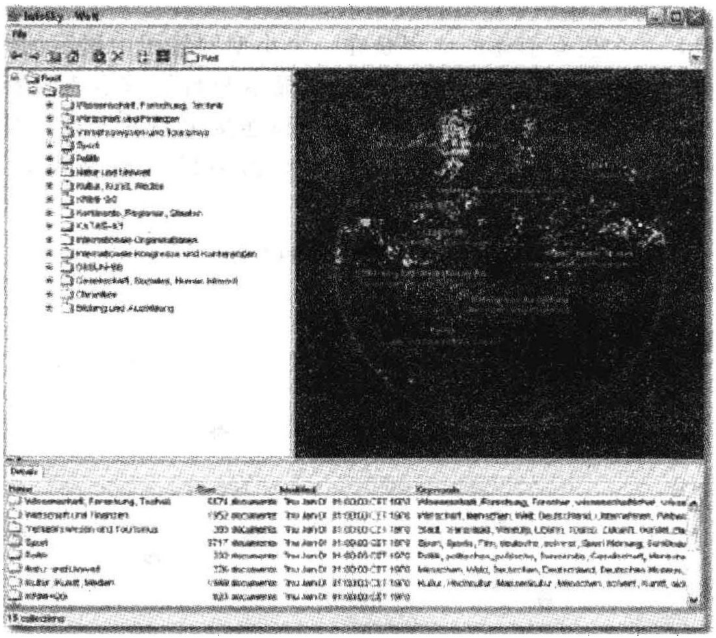


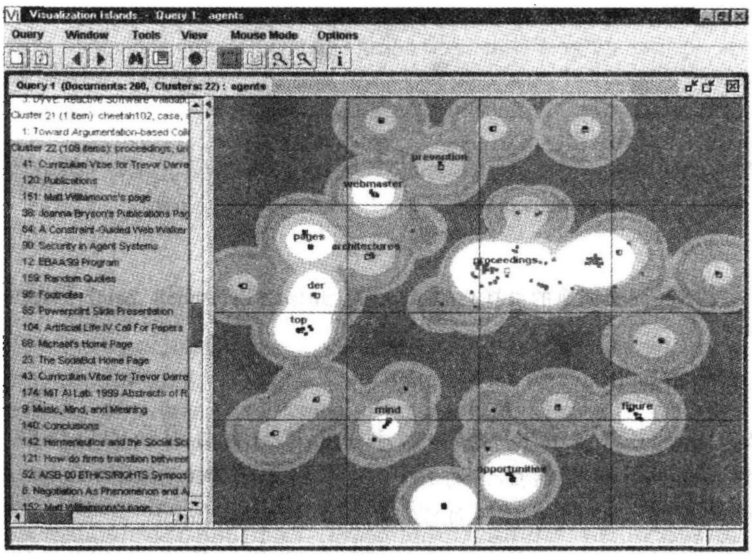
图 2-19 VIBE 的显示，其中查询项在二维空间内展示，而文档根据其文本来排列，源自 [1228]

这种想法的另一个形式是将文档或词语从一个高维的词项空间映射到二维的平面，然后文档或词语通过 2D 或 3D 显示在这个平面上 [53，662，758，1688，1700]。这种形式的聚

类能够用在根据查询检出的文档，或者在一个预处理过的文档集内加亮显示与查询相匹配的文档。图 2-20 是两个基于这种星空（starfield）理念的显示结果。



a）



b）

图 2-20　用一个 2D 或 3D 映射的标志符号来表示文档的想法已经提出了多次。这里显示两个例子：a）InfoSky，源自论文 Evaluating a system for interactive exploration of large，hierarchically structured document repositories，*Proceedings of the IEEE Symposium on Information Visualization*，pp. 127-133（Granitzer，M.，Kienreich，W.，Sabol，V.，Andrews，K. and Klieber，W. 2004），© 2004 IEEE [662]；b）xFind 中的 VisIslands，源自论文 Search result visualisation with xFIND，*Proceedings of User Interfaces to Data Intensive Systems*，pp. 50-58（Andrews，K.，Gutl，C.，Moser，J.，Sabol，V. and Lackner，W. 2001），© 2001 IEEE [53]

这些视图都很容易计算并且能够直观地显示。然而，现有的评价对它们的可用性提出了
47~48 负面的意见［662，773，910，1400］。主要的问题是，在这些视图中文档的内容是不可见的，而且2D表示不像语义那样可以进行复杂的交流。

利用这种思想，一个更有前途的应用是在一个小型网络图中展示词典中的索引项，例如 Visual Wordnet（见图 2-21)。这种方法通过只显示与目标节点直接相连的节点从而使较大的 WordNet 数据库得到简化。对于这种节点和连接关系的视图，其应用情况还没有在已发表的研究中进行过评价，但它在用于组织搜索结果时并没有获得很好的结果［1548］。

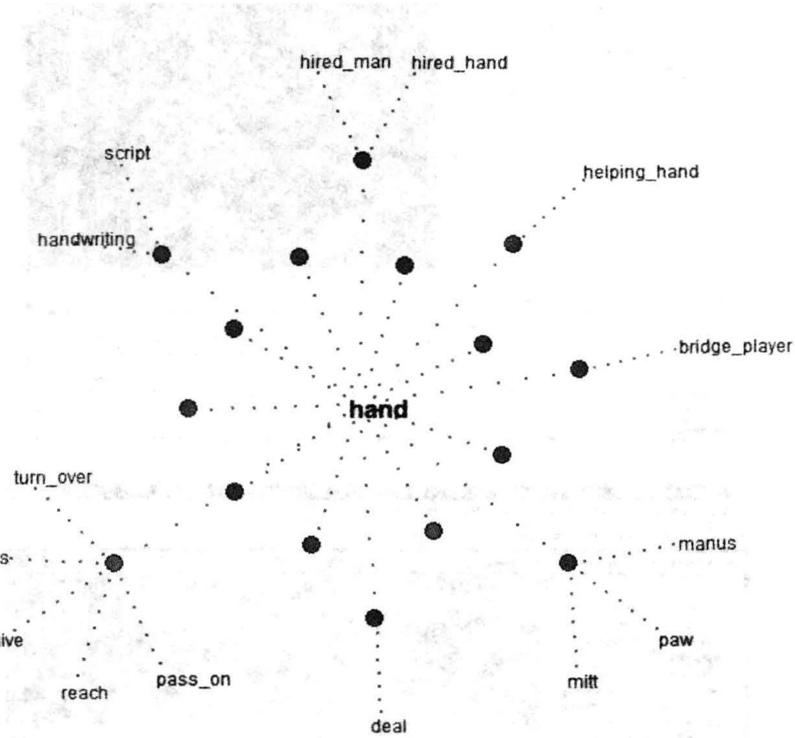


图 2-21 WordNet 同义词典的可视化表示，来自 http://kylescholz.com/projects/wordnet/

2.4.4 文本挖掘的可视化

2.4.3节说明了对于搜索结果来说，搜索可视化并没有很强的可用性。事实上，可视化似乎对文本数据的分析和探索更为有用。大多数搜索系统的用户对文档中的词语如何分布以及文档集中最常见的词语并不感兴趣，但这些是计算语言学家、分析学家以及奇特词语爱好者感兴趣的活动。像 Word Tree 那样的可视化系统［1669］会显示一部分文本词汇索引，使得用户能够看到哪些词语和短语会常常出现在给定词语的前后（见图 2-22)，另外还有 NameVoyager 系统［1670］，它显示在不同的年代中，美国婴儿名字的出
49 现频率（见图 2-23)。

对搜索界面进行可视化，有时是为了方便信息分析师。图 2-24 显示的是 TRIST 信息“分类”系统［854，1303］，它的作用是帮助信息分析师完成工作。系统将搜索结果用文档图标表示；数以千计的文档可以显示在一起，系统支持多维链接，从而使我们能够发现文档间的特征与相关性。图标的颜色用做显示哪些文档是用户之前已经看过的，图标的大小和形状分别表示文档的长度和类别。这看起来是个有效的系统，它的设计者连续两年在

IEEE 视觉分析科技竞赛（IEEE Visual Analytics Science and Technology，VAST）中获胜[679]。



图 2-22　Word Tree 可视化系统在 Martin Luther King 的演讲 "I have a dream"（我有一个梦想）上的演示，来自 The word tree，an interactive visual concordance，*IEEE Transactions on Visualization and Computer Graphics*，14（6），pp. 1221-1228（Wattenberg，M. 和 Fernanda，B.，2008），© 2008 IEEE [1669]

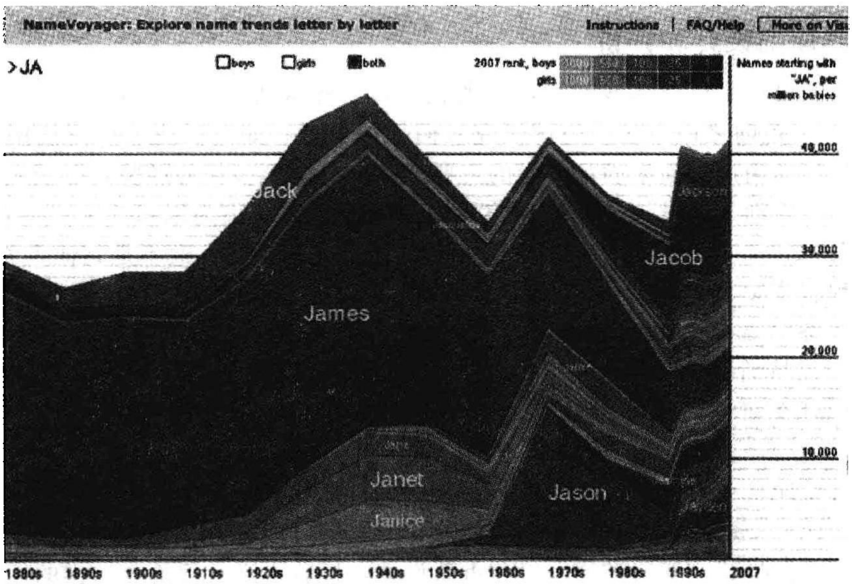


图 2-23　一个可视化演示的例子，针对一段时期内以 JA 开头的婴儿名字的相对普及度，来自 babynamewizard. com

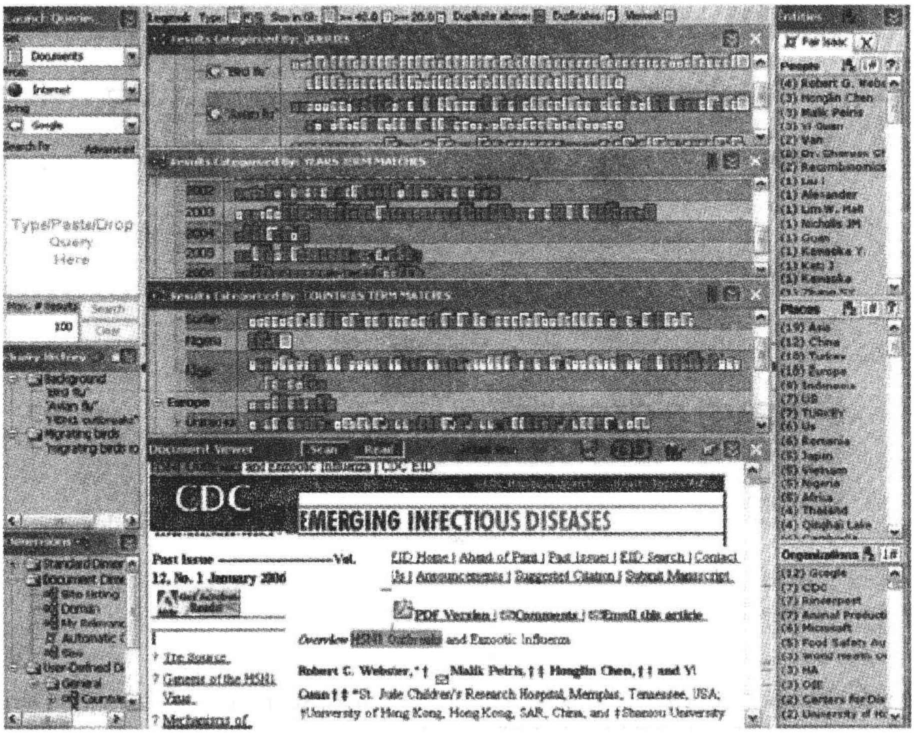


图 2-24 输入 Avian Flu 相关查询之后，TRIST 界面的显示，来自 Avian Flu case study with nSPace and GeoTime，*Proceedings of the IEEE Symposium on Visual Analytics Science and Technology* （*VAST* ' 06），pp. 27-34 （Proulx，P. et al. 2006），© 2006 IEEE [1303]

2.5 搜索界面的设计和评价

用户界面的设计是一项实践活动，它的技术包含在人机交互（Human-Computer Interaction，HCI）的范畴之内。这个领域研究人们如何思考、如何反应、如何使用技术，以及如何设计出最好的用户界面来满足人们的需要和倾向。基于多年的经验，已经制定出一些实践和指导方针，帮助人们设计成功的用户界面。这些实践统称为以用户为中心的设计（user-centered design），它围绕着用户的行为和思考过程完成设计，而不是其他无关的因素。

设计过程首先要预期用户的目标（goal）是什么，然后设计一个界面来帮助用户通过
50 完成一系列任务（task）来完成目标。在信息获取领域，目标的范围可能会非常广泛，从寻找管道工到保持对商业竞争对手的关注，从写作可发表的学术论文到调查一宗欺诈指控。信息获取任务可以用来完成这些目标。这些任务覆盖了从询问具体问题到彻底研究某个主题。

用户界面设计是一个不断改进的过程，其中的目标和任务通过对用户的研究来说明，然后构建初始设计——这通常会基于现存的设计，但也有可能包含一些新的想法。这些初始设计由预期用户进行测试，然后进行评价并重新设计，接着再进行评价，这样的循环需要重复很多次。

评价用户界面的过程通常与评价排序算法或爬虫技术不同。爬虫可以通过一些硬性的量化指标，如覆盖度和新鲜度来评价。排序算法可以通过精度、召回率和速度来评价。但是，用户界面的质量是由用户对它的反应来决定的。与量化指标相比，主观反应即使不能说是更

重要的，至少也是一样重要的，因为如果用户需要在两个系统中选择一个，那么他们会使用他们偏爱的那个。偏爱的原因可能是由多个因素决定的，包括速度、熟悉度、审美观、偏好特性，或者主观感知的排序准确性。通常更受青睐的选择会是用户熟悉的那个。尤其是在搜索界面中，一个新的界面必须要在主观感知上比旧的质量更好，用户才会转而使用新的界面。这一现象可以解释为什么自从搜索引擎第一次出现之后搜索结果的显示方法就没有什么明显的变化。

如何能够最好地评价用户界面，取决于当前处于开发周期中的哪个阶段。当开始一个新设计或新想法的时候，通常会使用简易（discount）可用性的方法。简易方法会向一些潜在的用户显示若干种不同的设计，然后让他们指出哪些部分是有前景的，而哪些是没有前景的。通常在这个任务中会使用草图上的原型设计，因为它们可以快速地开发，也很容易舍 51
弃，因为有证据显示，比起完整的成品，用户更愿意去批评那些明显未完成的东西［163，1344］。通常这个设计-测试-再设计的过程，在找到一个可接受的交互原型起始点之前会循环若干次。

接下来，应当进行非功能性的交互设计开发，并由少量的参与者进行测试，获取他们的主观反应，并确定哪些元素在设计中有良好的效果，以及哪些因素会产生混淆或者效果不佳。如果一个参与者看着屏幕而不知道该做什么，那么就意味着需要重新设计了。

另一种常见的折扣评价方法是*启发式评价*（heuristic evaluation），其中可用性专家“走查”（walk through）设计，然后评价其是否符合设计准则。这种评价方法在寻找可用性问题时有很好的效果，因为经验丰富的专家可以准确地预见潜在的问题，但是这一方法应当与目标用户的响应相结合。

在多次设计迭代后，通常就能设计出一个交互系统，此时可以在一个较正式的实验中进行测试，由研究的参与者来执行，并基于一系列的指标，将新的设计和一个有竞争力的基准进行对比，或者比较两种候选的设计方案。在评价搜索界面时，最重要的是让参与者有足够 52
的动力去完成那些任务［288，1524］。如果我们要求那些不关心照相机镜头或者对于照相机镜头非常了解的人去对照相机镜头做广泛的搜索，那么很可能不会产生有实际意义的结果。为了确保参与者的积极性，研究参与者应当对查询和信息集合的主题充满兴趣，另外应该选择那些会最终使用这个系统的人，或者接近的替代人员（例如，护理专业的学生通常要比执业护士更愿意测试一项设计）。

正式的实验通常旨在产生研究结果，用于发表以及在更为广泛的群体中应用，也有一些机构会在其内部进行正式的研究。正式的研究需要遵循科学领域中的实践方法，如招募研究参与者，对控制条件和实验条件进行比较。正式的实验应当谨慎地进行设计，需要考虑到那些潜在的干扰因素，比如要平衡那些参与竞争的设计的显示顺序，以及实验人员要避免对某个设计表现出偏见。只有达到这样要求的研究才能回答诸如“某个设计元素的优劣”或者“新的特性是否比现有的系统更好”之类的问题［735］。

这种研究能够发掘重要的主观性结果，如新的设计是否要明显好于基准系统，但是搜索界面本身的特点使得我们难以通过一小部分参与者来精确地找出可量化的差异。这个困难是由很多因素决定的，比如任务和查询对于系统行为的巨大影响——在许多搜索系统的研究中，无论是交互式的还是批处理式的分析，任务上的差异都具体表现在系统和参与者上的差异。另一个问题是人们提交的搜索可以有很多不同的形式，以至于很难直接量化地比较它们的输出。最后，时间变量在某些时候对于评价交互式的搜索会话并不是一个合适的指标，因为让搜索用户在搜索过程中理解他们的主题能够带来很大的好处，但是这可能会比其他设计

方案花费更多的时间。

面对这些问题，最近几年有两种评价搜索界面的方法开始逐渐流行起来。一种是进行纵向研究，意思是参与者长时间地使用一个新界面，并监控和记录他们的使用情况 [518, 1471]。评价是基于日志中记录的客观指标以及对参与者的问卷调查和当面访谈。在某些情况下，一种新方法的优势只有在用户使用了一段较长的时间后才会有明显的感受，所以只有长时间的研究才能发掘这种优势。例如，文献 [862] 中的研究发现，随着时间的推移，用户会根据界面相应地改变他们的搜索模式，该研究还揭示了在怎样的情况下新的特性是有用的，而什么时候它们是不需要的。另外，开始很吸引人的界面（比如说令人印象深刻的图形）随着时间的推移有可能会变得让人厌倦，导致我们希望重新使用之前熟悉的界面。

在过去几年里流行的另一项主要评价技术是在使用率较高的网站上进行的大规模实验。这种方法经常称为*水桶测试*（bucket testing）、*A/B 测试*，或者*平行航班测试*（paral-
53 lel flights）[921]。一个每天接收到成千上万甚至是几百万个查询的搜索引擎可以进行以下的研究：随机选择一个用户子集，向他们展示新的设计，将他们的反应和同样随机选择出来的继续使用现有界面的控制组用户进行对比 [61, 922, 919]。这与正式的可用性研究不同，因为其中的参与者并不晓得自己参与了这项研究，网站会在被挑选出来的访问者没有了解和同意的情况下向他们显示新的设计（大多数网站的用户协议都允许这类服务）。

这种形式的研究通常能在 24 小时内完成，不过通常建议整个流程为 1～2 个星期。有些性能评价，如哪个链接被点击等，在这样的测试中是尤其有信息性的。例如，显示查询建议的界面可以和不显示建议的界面进行对比，并记录下查询建议被点击的频率。另一个例子是，在文本结果列表中插入多媒体结果的影响，可以通过周围链接点击情况的变化以及新信息被点击的频率来进行评价。将控制条件下和实验条件下的用户行为进行比较，有些时候，我们可以比较，对于相同的查询，分别处于这两种条件下的用户行为，因为用户的数量是如此之大。这种研究的一个潜在缺点是有些熟悉网站的控制组的用户很可能会在一开始对他们不熟悉的界面给出负面的反应，所以有些研究会在评价过程中考虑初始反应状态的因素。这项技术通常也没有考虑到主观信息的因素，很多实验会通过后续调查来研究主观的反应。

2.6 趋势和研究问题

本章介绍了很多关于提高用户搜寻信息时的人机交互体验的方法。这仍然是一个快速发展的领域，界面的进步很有可能会带来更好的搜索结果以及更有效的信息创建者和使用者。未来最重要的前进方向是社交搜索、移动搜索界面、多媒体搜索，以及面向自然语言的查询，Hearst [735] 曾对此有过详细的论述。

2.7 文献讨论

Hearst 的《Search User Interfaces》[735] 对本章介绍的主题进行了深入的讨论。另一本可供参考的关于信息搜寻行为的书是 Marchionini 的《Information Seeking in Electronic Environments》[1082]。Lesk 的《Understanding Digital Libraries》[1005] 也有对搜索界面的讨论。

也有很多关于 HCI 和界面设计的书，包括 Shneiderman 等人的《Designing the User Interface》[1472]，Kuniavsky 的《Observing the User Experience: A Practitioner's Guide to User Research》[948]。一本古老但依旧出色的关于用户界面评估的书是 Nielsen 的《Usability

Engineering》[1204]。

很多书都介绍了如何设计网站，而与本章内容最相关的书是 Morville 和 Rosenfeld 的《Information Architecture for the World Wide Web》[1157] 第 3 版，其中有两章内容与搜索有关。Kalback 的《Designing Web Navigation，Optimizing the User Experience》[863] 54
讨论的是网站导航的设计。另外，有许多网站都致力于可用性和用户界面设计的研究与讨论。

现在有很多非常好的书介绍如何利用信息的可视化进行设计，包括 Few 的《Now You See It：Simple Visualization Techniques for Quantitative Analysis》和《Information Dashboard Design：The Effective Visual Communication of Data》[562，563]，以及 Tufte 的《The Visual Display of Quantitative Information》[1605]，不过这些书并不是只关注文本或搜索的可视化。

更多关于 Web 搜索引擎界面以及网页内容可视化的参考文献将在 11.7 节中进行介绍。 55

信息检索建模

3.1 信息检索模型

信息检索中的建模是一个以产生排序函数为目标的复杂过程。该排序函数能根据给定的查询给文档打分。这个过程包含两项主要的任务：1）构想出表示文档和查询的逻辑框架；2）定义一个针对给定查询计算文档排名的排序函数。这个逻辑框架通常基于集合、向量，或者概率分布。它直接影响了文档排名的计算，这些排名然后被用来对给定查询所返回的文档进行排序。

考虑到排序可能是检索系统中最重要的过程，我们将广泛、深入地讨论信息检索建模。我们首先确立建模和排序之间的关系。然后，我们形式化地描述检索模型，并列出检索模型的分类体系。

3.1.1 建模和排序

如第2章所述，信息检索系统通常采用索引项来索引和检索文档。严格地说，一个索引项是一个关键词（或者一组相关联的词），可以独立地表达某种意思，通常扮演名词性的角色。从更广义的形式看，索引项可以是文档集内文档正文中的任何一个词。

基于索引项检索的主要优势是可以高效地实现，并且可以简单地用查询进行查阅。简单性是重要的，因为这减少了用户制定查询的精力。然而，仅用几个词来表达查询意图限制了
57 所能表达的语义。这样也无怪乎检索出的文档经常与用户的查询不相关。如果考虑到大部分用户没有受过如何合理制定查询的训练，那么这个问题会由于这些隐患而变得更糟。其直接的后果是搜索引擎的用户对于许多查询的返回结果感到不满。

为了检索出查询的答案，任何检索系统都要处理一个核心问题——预测哪些文档会被用户看做是相关的，哪些会被他们看做是不相关的。这个问题本来就困难。而且，由于不同的用户可能对于何为相关、何为不相关有各自的看法，因此这个问题自然地包含了一定程度的不确定性和模糊性。对于这种情况，系统要实现一个预测算法，希望该算法能和大部分用户对大部分查询与答案相关与否的看法相似。这个预测算法基本上就是排序函数，用来对检出文档建立一个简单的排序，排在前面的文档更有可能是相关的。所以排序函数是检索系统的核心。

排序算法是根据文档相关性这一概念的基本前提来执行的。关于文档相关性的不同前提条件会产生不同的检索模型。正如我们在这里讨论的，所采用的信息检索模型决定了什么是相关的、什么是不相关的（例如，系统所实现的相关性概念）。我们的讨论包括了过去数年中提出的各种关键的信息检索模型，为本书中其余大部分章节提供了概念性的基础。

3.1.2 信息检索模型描述

信息检索模型是由形成排序算法的基础前提决定的。我们的描述如下。

定义 一个信息检索模型是一个四元组 $[\boldsymbol{D}, \boldsymbol{Q}, \mathcal{F}, R(q_i, d_j)]$，其中

1）**D** 是文档集中文档的逻辑视图（或表示）的集合。

2）**Q** 是用户信息需求的逻辑视图（或表示）组成的集合。这些表示称为查询。

3）$\mathcal{F}$是一个对文档、查询及其关系建模的框架，例如，集合与布尔关系、向量与线性代数运算、样本空间与概率分布。

4）$R(q_i, d_j)$ 是排序函数，对查询表达式 $q_i \in \boldsymbol{Q}$ 和文档表达式 $d_j \in \boldsymbol{D}$ 赋予一个实数。排序函数定义了关于查询 q_i 的文档次序。

为了建立信息检索模型，我们首先考虑文档的表示形式和用户信息需求的表示形式。对于一篇文档，其表示形式可以是这篇文档内所有词语的子集，例如可以通过从文档中去除禁用词（例如冠词和介词）获得。对于查询，其表示形式可以是查询词的超集，例如可以通过对原始查询增加同义词获得。给定了这些表示形式之后，我们接着为它们构想建模框架。这个框架也要能提供构建排序函数的直观构想。例如，对于经典的布尔模型，这个框架由文档 58
集与在这组集合上的标准运算组成。对于经典的向量模型，这个框架由 t 维向量空间、查询和文档的向量表示形式，以及在这之上的标准线性代数运算组成。对于经典的概率模型，这个框架由词在文档和查询上的概率分布，以及贝叶斯（Bayes）定理组成。正如本章中一直讨论的，当这个框架构建好后，就需要确定一个排序函数。

给定一个查询和文档的表示形式，例如 q_i 和 d_j，排序函数 $R(q_i, d_j)$ 给查询 q_i 所对应的文档 d_j 赋予一个排名（一个实数）。这个过程如图 3-1 所示。注意，在我们的讨论中，我们使用符号 q_i 表示查询及其表示形式，用符号 d_j 表示文档及其表示形式。

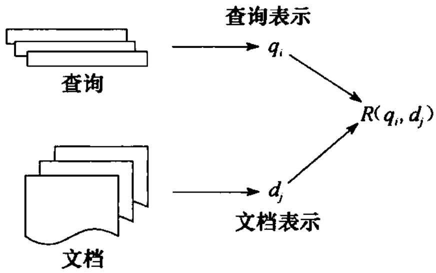


图 3-1 排序函数 $R(q_i, d_j)$ 用查询和文档的表示形式作为输入，给文档 d_j 赋予关于查询 q_i 的排名

在本章的其余部分，我们讨论各种不同的信息检索模型。在讨论中，我们不会显式地举出每个模型的成分：**D**、**Q**、$\mathcal{F}$和 $R(q_i, d_j)$。这些成分在讨论中是相当清楚的，并可以容易地推导得到。

3.1.3 信息检索模型的分类体系

信息检索模型主要基于文本，即它们根据文档的正文来对查询的相关文档进行排序。然而，在 Web 中，也需要使用网页上的链接结构来获得好的排序。另一方面，多媒体对象和正文文档编码的方法很不一样。图像编码为像素的位图，视频对象编码为关于图像的时间流，语音对象编码为关于声音的离散流。由于其表示形式的特殊性，多媒体对象的排序与文本很不一样，或者说检索时不用排序。针对这些特点，我们区别三种主要的信息检索模型：基于文本、基于链接和基于多媒体对象的模型。

图 3-2 说明了我们对信息检索模型和多媒体检索方法的分类体系。对于基于文本的模型，我们将其分为用于无结构文本的模型和考虑文本结构的模型。在第一类中，文本仅仅建 59
模为词语的序列。在第二类中，文本的结构化成分，例如标题、节、子节和篇章，是模型整体的一部分。因为也包含了无结构的文本，所以这些通常称为半结构。对于无结构文本，信息检索中的三个经典模型是布尔模型、向量模型和概率模型。在布尔模型中，文档和查询表示成索引项的集合。这样，如［683］提出的，我们说这个模型是基于集合论的。在向量模型中，文档和查询表示成 t 维空间里的一个向量。因此，我们说这个模型是基于代数的。在概率模型中，用来对文档和查询建模的框架是基于概率论的。因此，我们说这个模型是基于概率论的。

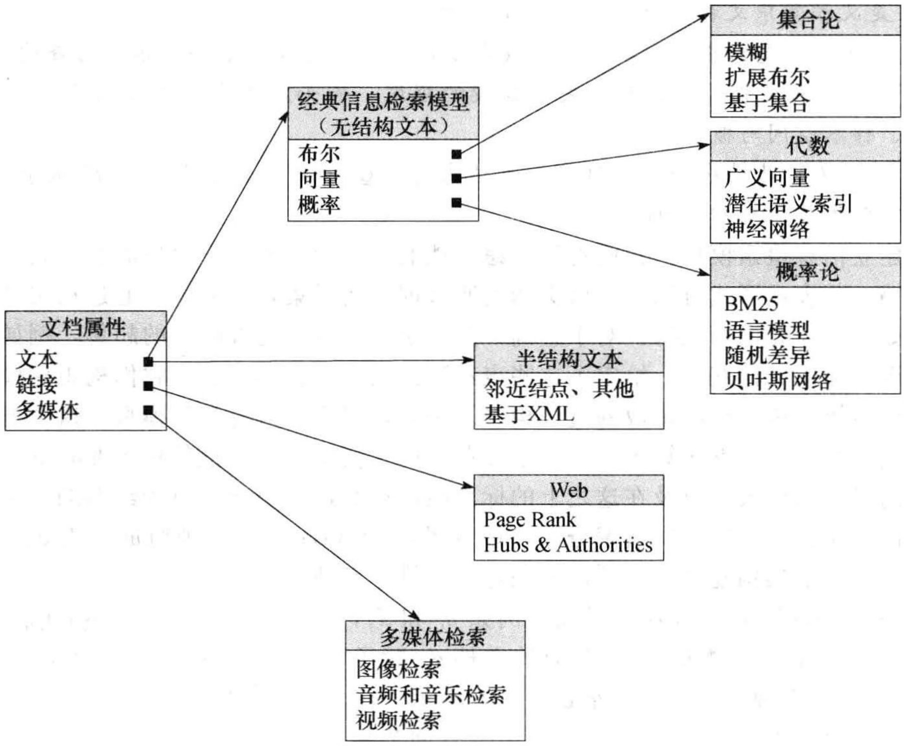


图 3-2 IR 模型的分类体系

多年来，研究人员已经提出了基于经典模型（即集合论、代数和概率论）的其他检索模型。对于其他的集合论模型，我们将其分为模糊、扩展布尔和基于集合的模型。其他的代数模型，我们将其分为广义向量、潜在语义索引和神经网络模型。对于其他的概率模型，我们将其分为 BM25、贝叶斯网、随机差异模型和语言模型。所有这些模型在本章中都详细
60 讨论。

对于半结构化文本检索模型（即考虑文本中的结构），我们考察索引方法，例如邻近结点和基于 XML 的索引方法。这些都涵盖在第 13 章中。

在 Web 中，由于文档（或者网页）数量巨大，只基于文本的排序是不够的，所以有必要考察将网页之间的链接作为模型的组成部分。这些促进了基于链接的检索模型的发展，尤其是我们将在第 11 章中讨论的 PageRank [263] 和 Hubs & Authorities [911]。

多媒体数据的检索采用一套截然不同的检索策略。例如，假设对于一幅图像，它表示为一个位的集合，用来描述像素的颜色和亮度，这些信息对于理解用户可能需要怎样的图像没有直接的关系。为了检索出让用户感兴趣的图像，有必要采取一系列搜索文档集所使用不到的中间步骤。例如，用户可能不是写出查询，而是提交一幅给定的图像来表达信息需求。这幅查询图像可以和数据集中的图像进行比较来找到相关的图像。这个方法是非常不稳定的，因为像素级上的共性可能有很大的误导性。有许多更复杂的方法来处理这个问题，正如第 14 章中所讨论的。重要的是，对多媒体检索的方法与文本的信息检索模型相差甚远。举例来说，许多多媒体检索模型不包含任何形式的排序。因此，它们在图 3-2 中被单独列出来，我们称之为**检索策略**。

多媒体检索最简单的形式是图像检索，因为图像是静态的。对于音频和视频来说，多媒体对象的表达形式也必须包含时间维度，这使得这些文件更加巨大，问题更加困难。图像、

音频和视频在第14章中详细讨论。

由于信息检索模型的主要目的是产生一组和用户相关的答案，因此现代信息检索系统的实现方案包含了不同信息检索模型的特征，而不仅是一种。例如，如今的Web排序函数结合了经典的信息检索模型和基于链接模型的特征来提高检索性能。

3.2 经典信息检索

在本节中，我们列出信息检索中的三种经典的模型，也就是布尔模型、向量模型和概率模型。在布尔模型中，索引项不带任何权重，它们仅是集合中的元素。在向量模型和概率模型中，索引项有权重，用于提高排序质量。

3.2.1 基本概念

1. 索引项或者关键词

信息检索中的经典模型认为每篇文档应描述为一个集合，该集合包含具有代表性的关键词，这些词称为索引项。

定义 一个索引项是文档中的一个词或者一组连续的词。索引项的最普遍形式是集合中任意一个词。这是搜索引擎设计人员采用的方法。从更严格的意义上说，索引项是一组预选的词，表示文档中的关键概念或主题。这是图书馆员和信息科学家采用的方法。 61

举例来说，一组预选的索引项可以用来概括文档的内容。在这种情况下，它们主要是名词，或者名词组，这是因为名词本身含有意义。形容词、副词和连词作为索引项用处不大，它们主要作为补语。然而，如果需要匹配文档中任意的词或者词语序列，那么有必要考虑把集合内所有不同的词作为索引项。Web搜索引擎采用这种方法，我们在第11章中讨论。

定义 假设 t 是文档集中索引项的数量，k_i 是某个索引项。$V=(k_1, k_2, \cdots, k_t)$ 是文档集中所有不同索引项的集合，通常称为文档集的词汇表 V。词汇表的大小是 t。

词汇表是文档集的一个重要组成部分，它确定了所有的索引项。随着文档集的增长，词汇表的大小也会增长，这是由拼写错误、不同形式的数字、各种不同的ID符号和首字母缩写词这些因素造成的。在Web中尤其如此，我们会在第11章中讨论。

2. 文档和查询的表示形式

文档和查询在信息检索系统中通过文档和查询的表示形式来建模，如图3-1所示。我们现在把这个概念形式化。

定义 如前所述，设 $V=(k_1, k_2, \cdots, k_t)$ 是文档集的词汇表。如果有三个索引项 k_l、k_m 和 k_n 出现在同一篇文档 d_j 中，我们说观察到索引项共现模式 $[k_l, k_m, k_n]$。对于一个大小为 t 的词汇表 V 来说，在文档集内出现的索引项共现模式的总量是 2^t。举例来说，模式 $(1, 0, \cdots, 0)$ 表示仅有索引项 k_1 出现而没有其他项。模式 $(1, 1, \cdots, 1)$ 表示所有的索引项均出现。每一个索引项共现模式称为一个索引项合取分量。对于文档 d_j，我们给予一个索引项合取分量 $c(d_j)$，用以描述哪些索引项出现在文档中，哪些没有。类似地，对于一个查询 q，我们给予一个索引项合取分量 $c(q)$，用以描述哪些索引项出现在查询中，哪些没有。索引项合取分量 $c(d_j)$ 提供了文档 d_j 在系统中的表示形式，索引项合取分量 $c(q)$ 提供了查询 q 在系统中的表示形式。

在这种情况下，查询和文档仅仅由索引项合取分量表示，它反映了它们包含的索引项。这是可采取的最简单的表示形式，经常称为词袋法（bag of words）。正如我们之后将看到

的，信息检索模型需要采用更加复杂的查询和文本表示形式来改善检索结果。

3. 项-文档矩阵

索引项在文档中的出现建立了索引项和文档之间的关系。这些项-文档关系能被量化，62 例如，索引项在文档中出现的频率。以矩阵的形式可以写为

$$\begin{array}{c} \\ k_1 \\ k_2 \\ k_3 \end{array} \begin{array}{c} \begin{array}{cc} d_1 & d_2 \end{array} \\ \begin{bmatrix} f_{1,1} & f_{1,2} \\ f_{2,1} & f_{2,2} \\ f_{3,1} & f_{3,2} \end{bmatrix} \end{array}$$

其中，每一项 $f_{i,j}$ 表示索引项 k_i 在文本 d_j 中出现的频率。相比于记录这个索引项是否出现在文档中，使用出现频率来量化项-文档之间的关系可以提供更多的信息。当然，这是一种简单的方法，更复杂的方法将在我们引入了项权重的概念后再讨论。

4. 文档的逻辑视图

索引项可以直接从文档的正文中抽取出来，也可以由图书馆员和信息科学家等专家来确定。无论这些具有代表性的索引项是自动生成的，还是由某个专家建立的，它们都提供了一种*文档的逻辑视图*。

现代计算机使得一篇文档可以用其包含的所有词来表示。这样的话，我们说这个检索系统采用了文档的全文文本逻辑视图（或者表示形式）。然而，对于规模巨大的文档集，人们可能关注如何减小代表性关键词集合的规模。这可以通过*禁用词*去除（例如冠词和连词）、*词干提取*（把不同的单词简化为它们共有的语法根形式）和名词组识别（这可去除形容词、副词和动词）的操作来达到。而且，还可以运用压缩。这些在正文上的操作（也称为文本转换）在6.6节中会详细讨论。它们减少了文档表示形式的复杂性，并把逻辑视图从全文转化为*索引项集合*。

全文是一个文档最完整的逻辑视图，但采用这种方式通常意味着更高的计算代价。由专63 家定义的一小组类别提供了一篇文档最精准的逻辑视图，为文档的表示添加了一个语义层。然而，用人工生成的类别来建索引通常会影响召回率，并可能导致一个糟糕的搜索体验，尤其当用户不是熟悉这个文档集的专家时。如图3-3所示，信息检索可以采用多个文档的中间逻辑视图。除了采用其中的某种中间表示形式外，检索系统也会识别出文档通常具有的内部结构（例如，章、节、子节）。也就是说，我们可以把逻辑化表示文档的问题看做一个（离散的）连续统一体，其中文本的逻辑视图（平滑地）从全文的表示形式转变为由专家定义的主题所组成的更精准的表示形式。

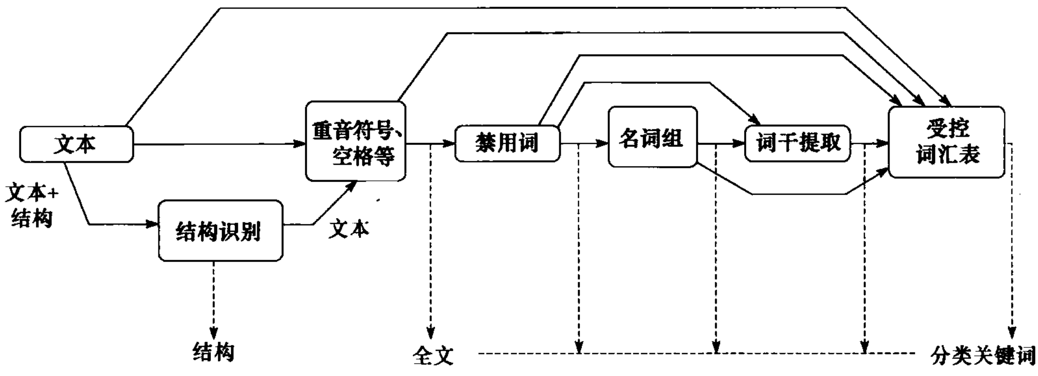


图3-3 文档的逻辑视图：从全文到索引项集合

上述的定义提供了讨论三种经典信息检索模型的基础，也就是现在我们要讨论的布尔模型、向量模型和概率模型。

3.2.2 布尔模型

布尔模型（Boolean Model）是一个基于集合论和布尔代数的简单模型。因此，该模型非常直观并有准确的语义含义。考虑到其固有的简洁形式，布尔模型过去受到了大量的关注，并被许多早期的商业文献目录系统采用。

布尔模型考察的是索引项是否出现在文档中，也就是说项-文档矩阵中的项-文档频率都是二值的。查询 q 由三种操作符所连接起来的索引项构成：非（*not*）、与（*and*）、或（*or*）（注意：考虑到补运算是代价昂贵的，出于效率的原因，系统可能选择使用一个“减”操作，而不是“非”操作）。这样，从本质上看，查询是传统的基于索引项的二值表示，如下所示。

定义　*在布尔模型中，所有的项-文档矩阵的元素或者是 1，表示这个索引项出现在这个文档中；或者是 0，表示这个索引项不出现在文档中。查询 q 是一个基于索引项的布尔表达式，例如，$[q = k_a \wedge (k_b \vee \neg k_c)]$。对于给定查询，满足其条件的索引项合取分量称为查询合取分量 $c(q)$。综合所有的查询合取分量，我们可以把查询重写为一个由这些合取分量组成的析取表达式。这称为查询析取范式，我们用 q_{DNF} 表示。*

举例来说，重新考察查询 $[q = k_a \wedge (k_b \vee \neg k_c)]$，并假定文档集的词汇表是 $V=\{k_a, k_b, k_c\}$，查询 q 可以写做析取范式 $[q_{DNF}=(1,1,1) \vee (1,1,0) \vee (1,0,0)]$，如图 3-4 所示。现在考虑包含索引项 k_a 和 k_c，但不包含 k_b 的文档 d_j。它的索引项合取分量 $c(d_j)$ 是（1，0，1），不是 q_{DNF} 的成分，不满足查询要求。所以我们说文档 d_j 不满足查询 q。

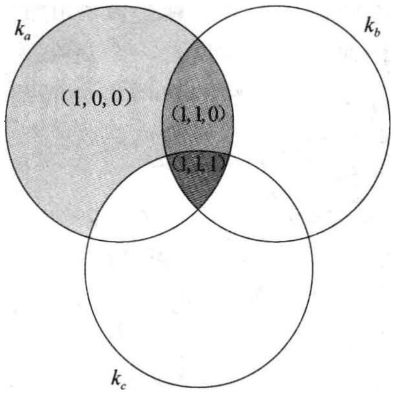


图 3-4　查询 $[q=k_a \wedge (k_b \vee \neg k_c)]$ 的三个合取分量

即使当文档集的词汇表包含不在查询中的词，这种方法也可行。举例来说，考察当词汇表是 $V=\{k_a, k_b, k_c, k_d\}$ 时，仅包含索引项 k_a、k_b 和 k_c 的文本 d_j 表示成索引项合取式（1，1，1，0）。而且，查询 $[q=k_a \wedge (k_b \vee \neg k_c)]$ 表示成析取范式

64

$$q_{DNF} = (1,1,1,0) \vee (1,1,1,1) \vee (1,1,0,0) \vee (1,1,0,1) \vee (1,0,0,0) \vee (1,0,0,1)$$

也就是说，当索引项 k_d 没有在查询中定义时，该项是否存在于合取分量中都被考虑到了。因此，满足查询的条件不变。若文档满足包含查询项的条件，那么就存在一个查询合取分量可以和文档合取分量匹配。简单起见，在本书中，我们把合取分量的表示形式限制在明确出现在查询中的索引项上。

定义　*在布尔模型中，查询 q 是传统的基于索引项的布尔表达式。设 $c(q)$ 是任意的查询合取分量。给定文档 d_j，设 $c(d_j)$ 是对应的文档合取分量。那么文档 d_j 和查询 q 之间的相似度定义为*

$$sim(d_j, q) = \begin{cases} 1 & \exists c(q) \mid c(q) = c(d_j) \\ 0 & \text{其他} \end{cases} \tag{3-1}$$

若 $sim(d_j, q)=1$，那么布尔模型断定文档 d_j 和查询 q 是相关的（事实上可能不是这样）；否则，断定文档是不相关的。

布尔模型断定每篇文档要么相关，要么不相关，没有部分相关的说法。例如，设 d_j 是一篇文档 $c(d_j)=(1, 0, 1)$。文档 d_j 包含了索引项 k_a，但是认为和查询 $[q=k_a \wedge (k_b \vee \neg k_c)]$ 不相关。这种没有分级度量概念的二元决策标准无法取得良好的检索质量。而且，尽管布尔模
65 型有准确的语义含义，但是把信息需求转化为布尔表达式通常并不容易。实际上，大部分用户觉得用布尔表达式表示查询需求既困难又别扭。结果，由用户编写的布尔表达式通常是非常简单的。

布尔模型的主要优点是其模型背后的简洁形式和其由于采用二值索引项权重带来的简单性。它主要的缺点是没有排序，这会导致检出太少或者太多的文档，并且对大部分用户而言，编写布尔查询是繁琐的。如今，大家都知道索引项权重可以带来检索质量的大幅提升，我们接着讨论。

3.2.3 项权重

给定某篇文档的索引项集合，我们注意到索引项在描述文档内容时不是同等重要。实际上，有些索引项比另一些不明确。确定索引项在多大程度上概括文本的内容并不容易。尽管困难，但索引项有一些性质是容易通过度量获得的，也有助于评价项的重要性。举例来说，考虑一个含有10万篇文档的文档集。在这10万篇文档中，把在每篇文档中都出现的词作为索引项是没有用的，因为这个词没有告诉我们关于用户对哪篇文档感兴趣的任何信息。相反，一个只在文档集中5篇文档中出现的词是非常有用的，因为它极大地缩小了用户关注的文档集合的范围。这样，大家就明白不同的索引项在用于描述文档内容时有不同的重要程度。这一效果可以通过给文档中每个不同的索引项赋一个数值权重来达到。

定义 *为了表征索引项的重要性，可以赋予文档集中每个文档 d_j 的索引项 k_i 一个权重 $w_{i,j}$，$w_{i,j}>0$。当索引项 k_i 不存在于文档中时，$w_{i,j}=0$。*

权重 $w_{i,j}$ 量化了索引项在描述文档内容方面的重要性。针对给定查询，对索引项赋权重可赋予每篇文档一个数值排名，从而改善检索结果。

为了生成索引项的权重，我们考察该项对于描述某一篇文档或者文档集中的一组文档有多大作用。我们用一个例子进一步说明。假设有一组关于 John Lennon 的文档，并假定有1000篇这样的文档。如果用户设定了查询“John Lennon”，我们怎样对文档排序，使用户对排在前面的文档更感兴趣？答案是，在没有额外信息的情况下，我们无法对文档进行排序。对于这个特别的集合，由用户提交的查询完全不包含任何有价值的信息。我们说关于查询的信息量是零。

这个独特的例子没有解释如何设计排序算法，但是清楚说明了关于索引项权重的一个重要观点，即不同的索引项不是同等重要的，应该有不同的权重。而且，权重受到文档集中文
66 档的影响。为了说明项权重的重要性，我们计算的权重必须反映索引项在文档集中的重要性，以及索引项在每篇文档中的重要性。这些权重依赖于索引项在文档中出现的频率，我们的定义如下所示。

定义 *设 $f_{i,j}$ 是索引项 k_i 在文档 d_j 中出现的频率，即索引项 k_i 出现在文档 d_j 正文中的次数。索引项 k_i 在文档集中出现的总频率 F_i 是该索引项在所有文档出现频率的总和。*

$$F_i = \sum_{j=1}^{N} f_{i,j} \tag{3-2}$$

其中，N 是文档集中文档的数量。索引项 k_i 的文档频率是包含该项的文档的数量，用 n_i 表

示。注意 $n_i \leqslant F_i$。

项间相关性

在我们的讨论中，假设索引项权重之间是相互独立的。这意味着知道二元组（k_i，d_j）的权重 $w_{i,j}$ 无法告诉我们二元组（k_{i+1}，d_j）的权重 $w_{i+1,j}$。这明显是一种简化，因为文档内出现的索引项之间不是毫无关联的。举例来说，假设索引项“计算机”和“网络”被用来对关于计算机网络的文档进行索引。在文档中，其中一个词的出现通常会跟着另一个词。这样，它们是相互关联的，并且其权重应该反映这种相关性。

考虑索引项间的相关性，我们能计算一个相关性矩阵，如图 3-5 所示。其中关键的概念如下所述。

定义　设 $\boldsymbol{M}=[m_{i,j}]$ 是一个 t 行 N 列的项-文档矩阵，其中 $m_{i,j}=w_{i,j}$，即矩阵中的每一元素 ij 由项-文档二元组（k_i，d_j）的权重给出。给定 $\boldsymbol{M}^{\mathrm{T}}$ 是矩阵 $\boldsymbol{M}$ 的转置，矩阵 $\boldsymbol{C}=\boldsymbol{M}\cdot\boldsymbol{M}^{\mathrm{T}}$ 是一个项间相关性矩阵。每一元素 $c_{u,v}\in\boldsymbol{C}$ 表达了索引项 k_u 和 k_v 之间的关系，如下所示。

$$c_{u,v}=\sum_{d_j} w_{u,j}\times w_{v,j}$$

项间相关性矩阵 $\boldsymbol{C}$ 建立了任意两个索引项 k_u 和 k_v 之间的关系，如图 3-5 所示，这是基于它们在文档集中联合出现的情况确定的。这个关系由相关系数 $c_{u,v}$ 来量化。文档中索引项 k_u 和 k_v 共现的次数越高，它们的相关性就越高。而且，我们能利用文档内索引项之间的距离来改进项间相关性的估计。其目的是文档中距离相近的项相比于距离远的项有更强的联系。更详细的内容可参考第 5 章中关于计算项间相关性的三种不同方法。 67

$$\underbrace{\begin{array}{c} \\ k_1 \\ k_2 \\ k_3 \end{array}\begin{array}{c} \begin{array}{cc} d_1 & d_2 \end{array} \\ \begin{bmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \\ w_{3,1} & w_{3,2} \end{bmatrix} \\ \boldsymbol{M} \end{array} \quad \times \quad \begin{array}{c} \\ d_1 \\ d_2 \end{array}\begin{array}{c} \begin{array}{ccc} k_1 & k_2 & k_3 \end{array} \\ \begin{bmatrix} w_{1,1} & w_{2,1} & w_{3,1} \\ w_{1,2} & w_{2,2} & w_{3,2} \end{bmatrix} \\ \boldsymbol{M}^{\mathrm{T}} \end{array}}$$

$$\Downarrow$$

$$\begin{array}{c} \\ k_1 \\ k_2 \\ k_3 \end{array}\begin{array}{c} \begin{array}{ccc} k_1 & k_2 & k_3 \end{array} \\ \begin{bmatrix} w_{1,1}w_{1,1}+w_{1,2}w_{1,2} & w_{1,1}w_{2,1}+w_{1,2}w_{2,2} & w_{1,1}w_{3,1}+w_{1,2}w_{3,2} \\ w_{2,1}w_{1,1}+w_{2,2}w_{1,2} & w_{2,1}w_{2,1}+w_{2,2}w_{2,2} & w_{2,1}w_{3,1}+w_{2,2}w_{3,2} \\ w_{3,1}w_{1,1}+w_{3,2}w_{1,2} & w_{3,1}w_{2,1}+w_{3,2}w_{2,2} & w_{3,1}w_{3,1}+w_{3,2}w_{3,2} \end{bmatrix} \\ \text{项间相关性矩阵} \end{array}$$

图 3-5　在仅有两篇文档和三个索引项的小规模样例文档集上的项-文档矩阵、转置矩阵，以及项间相关性矩阵

鉴于项间相关性有多种计算方法，不同的信息检索模型尝试利用这一关系。其中包括广义向量模型、模糊信息检索模型、基于集合的模型和语言模型，我们将在后面的章节中讨论。在这种情况下，像大部分的信息检索模型那样，索引项的独立性假设似乎是一种过分的简化。然而，索引项独立性的确简化了项权重的计算，使得排序过程变快了。此外，利用索引项之间的相关性来提高最终的排序不是一项简单的工作。举例来说，人们对索引项独立性对于一般的文档集有多大用处没有共识。因此，除非清楚地说明，否则我们假设索引项是相互独立的。

对于项权重，主要的计算方法最早是由 Sparck Jones [1504] 和 Salton 和 Yang [1418] 提出。他们的工作出于经验性的实验，并获得了有效的称为 TF-IDF 的项权重框架。

3.2.4 TF-IDF 权重

项频（Term Frequency，TF）和反比文档频率（Inverse Document Frequency，IDF）是信息检索系统中最常见的项权重框架，称为 TF-IDF，我们接下来讨论。

1. 项频权重

定义 Luhn 假设。*索引项 k_i 在文档 d_j 中的价值，或者说权重，应该与项频 $f_{i,j}$ 成正*
68 *比。也就是说，项 k_i 在文档 d_j 的正文中出现越多，则其项频权重 $TF_{i,j}$ 也越高。*

这个假设基于以下的观察，即高频项对描述文档的关键主题是重要的，它直接引出如下的 TF 权重公式：

$$tf_{i,j} = f_{i,j} \tag{3-3}$$

在文献中采用的 TF 权重的变种是

$$tf_{i,j} = \begin{cases} 1 + \log f_{i,j} & f_{i,j} > 0 \\ 0 & \text{其他} \end{cases} \tag{3-4}$$

其中 log 以 2 为底。在这里，我们将采用 log 表达式作为 TF 权重的首选形式，因为这使得它们可以和 IDF 权重比较（也是 log 函数的形式，我们之后会讨论）。举例来说，如图 3-6 中的小规模样例文档集，对这个集合，项频和对数 TF 权重如表 3-1 所示。

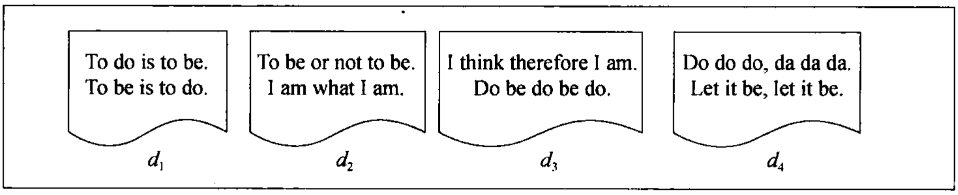


图 3-6 由四篇文档组成的小规模样例文档集

69 **表 3-1 图 3-6 样例文档集上的项频 $f_{i,j}$ 和对数 TF 权重 $TF_{i,j}$。文档大小是文档内词的个数，也在表中列出**

#	项	$f_{i,1}$	$f_{i,2}$	$f_{i,3}$	$f_{i,4}$	$TF_{i,1}$	$TF_{i,2}$	$TF_{i,3}$	$TF_{i,4}$
1	to	4	2	—	—	3	2	—	—
2	do	2	—	3	3	2	—	2.585	2.585
3	is	2	—	—	—	2	—	—	—
4	be	2	2	2	2	2	2	2	2
5	or	—	1	—	—	—	1	—	—
6	not	—	1	—	—	—	1	—	—
7	I	—	2	2	—	—	2	2	—
8	am	—	2	1	—	—	2	1	—
9	what	—	1	—	—	—	1	—	—
10	think	—	—	1	—	—	—	1	—
11	therefore	—	—	1	—	—	—	1	—
12	da	—	—	—	3	—	—	—	2.585
13	let	—	—	—	2	—	—	—	2
14	it	—	—	—	2	—	—	—	2
文档大小（# 词数）		10	11	10	12				

Salton 和 Yang [1418] 的实验证明了，在有些情况下，项频权重对于提高基于二值权重的检索系统是有用的。然而，他们观察到，当测试文档集和查询集改变的时候，其增益并

不一致。项权重需要更进一步地改进，这样 Sparck Jones 提出了反比文档频率。

2. 反比文档频率权重

通过反思生成文档集索引项的过程和将这些索引项用于检索的情况，Sparck Jones 构想出关于这种索引项特异性的统计解释，称为 IDF，它是项权重的基石。她的解释基于启发式探索，并激发了为 IDF 提供理论依据进行深入的研究工作。在这些解释中，一种有见地的观点，同时也以经典的概率模型阐述的 IDF 原理，是 Robertson 在［1371］中提出的。这里，我们根据由 Sparck Jones 基于语言项的穷尽性（exhausitivity）和特异性（specificity）的原始论点来讨论 IDF。

定义　穷尽性和特异性。*穷尽性是文档描述的一种属性，特异性是索引项的属性。文档描述的穷尽性可以解释为文档对其主要主题的覆盖度。索引项的特异性可以解释为这个索引项能多好地描述文档的主题。*

如果我们向文档中增加新的索引项，那么文档描述的穷尽性也会增加。而且，这个文档和一个输入查询相关的概率也会增加，即检索概率的提升。这自然引入了如下的最优穷尽性的概念。

定义　最优穷尽性。*赋给文档越多的索引项，文档描述的穷尽性就越大。从查询流中任意选出的查询检出文档的概率也增加了。然而，如果一篇文档被赋予了太多的索引项，那么它将被与它不相关的查询检出。这表明每篇文档索引项的平均数量需要优化，以最大化检出文档的相关概率。这个索引项的最优数量定义了文档描述的最优穷尽性。*

举例来说，像搜索引擎那样用一篇文档所有的项为其建立索引可能不是最好的方法。一种规避这个问题的方法是使用所有的项来索引，并根据其特异性设置权重，以区别不同索引项的重要性。

特异性是索引项的语义属性，即索引项或多或少是根据它的意义确定的。举例来说，项“tea”和“beer”比“beverage”更具体。这样，如果索引是手动完成的，那么相比于“tea”和“beer”，我们可以期望索引项“beverage”可用来为更多的文档索引。这个关于索引项特异性的解释是基于索引项描述文档主题的准确性，是信息科学研究人员广泛采用的。 70
另一种方法是把特异性看做索引项使用情况的函数。也就是把索引项的特异性解释为统计属性而不是语义属性，如下所示。

定义　统计上的索引项特异性和文档穷尽性。*在统计上，文档描述的穷尽性可以被其包含的索引项量化。而且，索引项的特异性可以被该项出现的文档数量的反函数所量化。*

穷尽性和特异性之间的关系现在清楚了。若文档描述变得越长，索引项的特异性就变得越低。尤其是，若某个索引项出现在文档集中所有的文档中，那么它的特异性最小，且该项对检索无用。这样的逻辑自然会产生由特异性确定项权重的想法。对此，项权重可以表达为项出现的相对（relative）频率，如下所示。

在自然语言的正文中，词语的相对文档频率可以由数学概率分布来估计，通常称为 Zipf 定律［1793］，在 6.5.2 节中有详细的讨论。

定义　Zipf 定律。*如前所述，设 n_i 是索引项 k_i 的文档频率。对所有索引项的文档频率按降序排列，并设 $n(r)$ 是第 r 个项的文档频率。我们说 r 是文档频率为 $n(r)$ 的索引项的序号。那么，根据 Zipf 定律，*

$$n(r) \sim r^{-\alpha} \tag{3-5}$$

其中，α 是由经验常数。也就是说，项的文档频率可以建模为其序号的指数函数。

Zipf 定律的等式也可以写为

$$n(r) = Cr^{-\alpha}$$

其中 C 是另一个由经验常数。对英语来说，$\alpha=1$ 提供了一种简单的近似。设 $\alpha=1$ 并采用 log 函数（我们以 2 为底，除非另加说明），

$$\log n(r) = \log C - \log r$$

这是基本的幂律方程（即，在 log-log 尺度变换下，变量和其序号之间的 Zipf 关系是负线性的）。若 $r=1$，我们有 $C=n(1)$，即 C 的值等于文档集中索引项的最大文档频率，并用做归一化系数。一种简化的方法是使用可能的最大文档频率来进行归一化，即采用 $C=N$，其中 N 是文档集中的文档数量，如前所示。这样，我们有

$$\log r \sim \log N - \log n(r)$$

我们可以用幂律给索引项赋权重，如下所示。设 k_i 是文档频率排在第 r 位的索引项，即 $n(r)=n_i$，则索引项 k_i 的权重 IDF_i 是

71

$$IDF_i = \log \frac{N}{n_i} \tag{3-6}$$

其中 IDF_i 称为索引项 k_i 的反比文档频率（因为 n_i/N 仅仅是相对文档频率）。

表 3-2 说明了图 3-6 的样例文档集中的 IDF 权重。我们观察到在文档集中最有区分性的索引项仅出现在一篇文档中，最没有区分性的索引项则出现在所有文档中。在一个大的真实文档集中，我们期望最具区分性的索引项是名词或者名词组（由一些词语组成的名词词组）。最没有区分性的项通常是冠词、连词和介词，这些通常称为禁用词（stop words）。

表 3-2 图 3-6 的样例文档集（$N=4$）上的 IDF 值

#	项	n_i	$IDF_i=\log(N/n_i)$	#	项	n_i	$IDF_i=\log(N/n_i)$
1	to	2	1	8	am	2	1
2	do	3	0.415	9	what	1	2
3	is	1	2	10	think	1	2
4	be	4	0	11	therefore	1	2
5	or	1	2	12	da	1	2
6	not	1	2	13	let	1	2
7	I	2	1	14	it	1	2

在 Sparck Jones 提出 IDF 权重之后不久，Salton 和 Yang 设计了实验来确认 IDF 在排序中的重要性 [1418]。如今，IDF 权重为现代索引项权重公式提供了基础，并被几乎所有的现代信息检索系统使用。

3. TF-IDF 权重

最流行的项权重公式是由 Salton 和 Yang 提出的 [1418]，它是把 IDF 因素和项频结合起来的权重计算方法。

定义 设 $w_{i,j}$ 是二元组 (k_i, d_j) 的项权重。那么，我们定义

$$w_{i,j} = \begin{cases} (1+\log f_{i,j}) \times \log \dfrac{N}{n_i} & f_{i,j} > 0 \\ 0 & \text{其他} \end{cases} \tag{3-7}$$

这称为 TF-IDF 权重公式。

表 3-3 说明了样例文档集中的 TF-IDF 权重。即使该表被简化过，但这个例子仍说明了 TF-IDF 权重公式的性质。索引项越稀少权重就越高，因为它们更有区分性。而且，在一篇

72

文档中出现越多的索引项就拥有更高的相对频率。

表 3-3 图 3-6 的样例文档集上的 TF-IDF 权重，用公式 3-7 计算得到。表中也列出了用带权向量的范数计算得到的文档长度

#	项	d_1	d_2	d_3	d_4
1	to	3	2	—	—
2	do	0.830	—	1.073	1.073
3	is	4	—	—	—
4	be	—	—	—	—
5	or	—	2	—	—
6	not	—	2	—	—
7	I	—	2	2	—
8	am	—	2	1	—
9	what	—	2	—	—
10	think	—	—	2	—
11	therefore	—	—	2	—
12	da	—	—	—	5.170
13	let	—	—	—	4
14	it	—	—	—	4
文档大小（向量范数）		5.068	4.899	3.762	7.738

尽管简单，但 TF-IDF 权重对一般的文档集是相当有效的，也就是说，可以给我们一无所知的新文档集中的索引项赋予权重。

4. TF-IDF 的变体

在参考文献中记述了 TF-IDF 权重表达式的好几种变体。其中最值得关注的是由 Salton 和 Buckley [1410] 提出的变体，以及在 Witten、Moffat 和 Bell [1709] 的文献中提出的变体，下面将详细讨论。

表 3-4 列出了 TF 权重的五种不同的变体。如果项出现在文档（或查询）中，那么二值（binary）变体赋予权重为 1；否则，为 0。原始项频（raw frequency）变体是一种直接使用项频的基本方法。对数归一化（log normalization）变体类似于 IDF 权重，采用了对数函数，当频率增加时，略微增加权重。两倍归一化 0.5（double normalization 0.5）变体引入了两个效应：首先，它用文档（或者查询）的最大频率对权重进行归一化；其次，它把权重归一化到 0.5～1 之间。两倍归一化 K 变体（double normalization K）只是一种泛化形式。通过变化 K，可以减少或者增加 $f_{i,j}$ 在 TF 权重中的影响。

表 3-5 列出了 IDF 权重的五种不同的变体。一元（unary）变体给所有索引项的 IDF 赋予 1，即剔除 IDF 因素。反比文档频率（inverse frequency）变体是我们上面讨论的标准公式。反比平滑文档频率（inverse frequency smooth）变体给对数的真数增加 1，避免当 n_i 出现极端情况下产生异常情形。反比最大文档频率（inverse frequency max）变体不是用文档集中文档的个数，而是用最大的文档频率来计算权重。概率反比文档频率（probabilistic inverse frequency）变体是由经典的概率模型变化得到的，将在 3.2.7 节中讨论。

表 3-4 TF 权重的变体

权重框架	TF 权重
二值	{0, 1}
原始项频	$f_{i,j}$
对数归一化	$1+\log f_{i,j}$
两倍归一化 0.5	$0.5+0.5\frac{f_{i,j}}{\max_i f_{i,j}}$
两倍归一化 K	$K+(1-K)\frac{f_{i,j}}{\max_i f_{i,j}}$

表 3-5 IDF 权重的变体

权重框架	IDF 权重
一元	1
反比文档频率	$\log\frac{N}{n_i}$
反比平滑文档频率	$\log\left(1+\frac{N}{n_i}\right)$
反比最大文档频率	$\log\left(1+\frac{\max_i n_i}{n_i}\right)$
概率反比文档频率	$\log\frac{N-n_i}{n_i}$

73

TF 变体和 IDF 变体之间不同的组合产生 TF-IDF 权重不同的形式。给定两个文档集C_1和C_2，对每个文档集最合适的 TF-IDF 权重形式可能不同。Salton 在 [1408] 中推荐的三种 TF-IDF 权重形式如表 3-6 所示。除非另行说明，本书的例子中采用第三种权重框架。

表 3-6 推荐的 TF-IDF 权重框架

权重框架	文档项权重	查询项权重
1	$f_{i,j} \times \log \frac{N}{n_i}$	$\left(0.5+0.5 \frac{f_{i,q}}{\max_i f_{i,q}}\right) \times \log \frac{N}{n_i}$
2	$1+\log f_{i,j}$	$\log\left(1+\frac{N}{n_i}\right)$
3	$(1+\log f_{i,j}) \times \log \frac{N}{n_i}$	$(1+\log f_{i,q}) \times \log \frac{N}{n_i}$

5. TF-IDF 的性质

为了更好地理解 TF-IDF 权重的行为，考察一个由 98 732 篇 1987—1989 年出版的
74 《Wall Street Journal》（华尔街日报）组成的参考文档集 [702]。可以通过如下的公式研究 TF、IDF 和 TF-IDF 权重在这个文档集上的行为。

$$TF_i = 1 + \log \sum_{j=1}^{N} f_{i,j} \tag{3-8}$$

$$IDF_i = \log\left(\frac{N}{n_i}\right)$$

注意到，为了在整个文档集上表达某个索引项的 TF 权重，我们把该索引项在所有文档中的项频都累加起来（如前定义的那样，这是文档集总项频 F_i）。设 r 是每个索引项根据 TF 权重的排名（即排名为 1 的项在文档集中有最大的 TF 权重）。图 3-7 说明了，对于《Wall Street Journal》（华尔街日报）参考文档集，式（3-8）定义的 TF 和 IDF 权重如何随着索引项排名的变化而变化的。我们观察到 TF 和 IDF 权重表现出的幂律特性会相互平衡。高 TF 权重趋于和低 IDF 权重联系在一起，而低 TF 权重趋于和高 IDF 权重联系在一起。结果，最大的 TF-IDF 权重是由具有中等 IDF 权重的索引项构成的。也就是说，在一个大的文档集中最具区分性的不是那些有最高 IDF 权重的索引项，而含有中等 IDF 权重的索引项具有最大的 TF-IDF 权重（这是一个用于排序的重要现象，首先由 Salton、Yang 和 Yu [1417] 提出）。换而言之，像禁用词那样的普通索引项和像外语单词或者错误拼写那样的稀少索引项对于排序都没有很大价值。

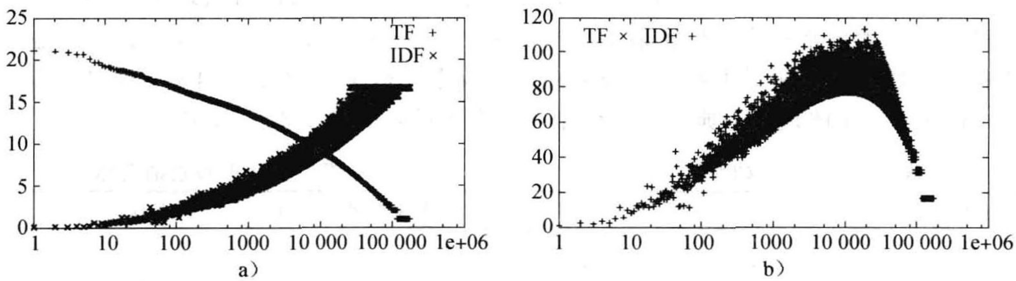


图 3-7 《Wall Street Journal》（华尔街日报）参考文档集的 TF、IDF 和 TF-IDF 权重，按照 TF 权重的降序排列（用对数尺度绘制）

3.2.5 文档长度归一化

在大的文档集中，文档长度可能变化很大。这是一个问题，因为对于给定的查询，较长

的文档仅仅因为它们含有更多的索引项而更可能被检出。为了抵消这一不良的影响，我们把每篇文档的排序除以其长度，这个过程通常称为*文档长度归一化*。因为这确实能导致更好的排序（即更符合用户相关性感知的排序），文档长度归一化被信息检索模型广泛地采用。 75

然而，为了恰当地归一化文本的排序，必须建立如何计算文档长度的方法。这可以通过多种方法来实现，取决于文档采用的表示形式。

定义　字节大小。*假设每篇文档都简单地用字节流表示。在这种情况下，文档长度是流中字节的数量，即文档的字节大小。这种表达方式的主要优点是其简单性。*

定义　词语数量。*假设每篇文档由一个单一的字符串表示，这个字符串可以分解成词语。在这种情况下，文档长度是其词语的数量。这种表达形式也简单，但是要在句法层计算文档长度，它包含了更多语义信息。*

定义　向量范数。*假设每个索引项被赋予 t 维空间中的一个正交单位向量 $\vec{k}_i$（其中，t 是索引项的总数）。在这个空间里，文档可以表示为如下的索引项带权向量。对于文档向量 $\vec{d}_j$ 的索引项 k_i，用向量 $w_{i,j}\times\vec{k}_i$ 表示该项对这篇文档的重要性。文档表达式 $\vec{d}_j$ 成为由所有索引项向量分量组成的向量，如图 3-8 所示。文档长度由向量范数给出，见式（3-9）。*

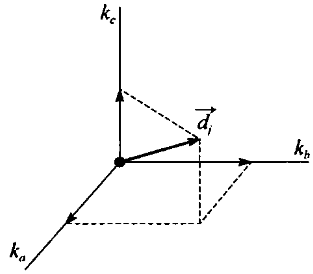


图 3-8　文档向量 $\vec{d}_j$

$$|\vec{d}_j| = \sqrt{\sum_i^t w_{i,j}^2} \tag{3-9}$$

把文档表示为字节流或者字符串，因为可获得文档长度的简单度量，即文档内字节的大小或词语的数量。带权重的向量表达式则考虑不同索引项的权重而导致更复杂的文档长度计算。 76

表 3-7 说明了图 3-6 的样例文档集中的文档长度。为了用字节计算文档大小，我们认为每行都有一个行结束符，每篇文档末尾都有一个文件结束符。并且，每一对连续的词语由一个空白字符分开。我们注意到把文档按照字节大小或者词语数量计算获得的文档长度之间的区别不超过 26%（最短和最长文档之间的相对区别）。然而，把向量范数作为文档长度计算会引入更多的变化，其中有两篇文档长度的相对区别超过了 100%。这是因为考虑了索引项权重，使得含有更多区分性索引项的文档被加强了，如样例文档集合中文档 d_4 的情况。

表 3-7　图 3-6 的文档集上三种文档长度的变体

长度	d_1	d_2	d_3	d_4
字节大小	34	37	41	43
词语数量	10	11	10	12
向量范数	5.068	4.899	3.762	7.738

文档长度归一化是被广泛采用的重要排序原则，主要原因是因为采用该方法普遍提高了结果的质量。仅仅因为更长，长文档更有可能和一个新查询的索引项相匹配，但这并不一定意味着文档和查询相关。相反地，很长的文档可能只是一个损害了检索结果的噪声。为了减小文档长度对文档排序的影响，最好的信息检索模型在排序中采用某种文档归一化方法。这对于长度变化很大的文档集尤为有用。

3.2.6 向量模型

向量模型 [1504，1418，1416] 认识到布尔匹配太有限，提出了一套可以进行部分匹配的框架。这是通过对查询和文档中的索引项赋予非二值权重实现的。这些权重最终用来计算系统中存储的文档和用户查询之间的相似度。通过对检出文档按相似度的降序排列，向量模

型考虑和查询仅有部分相匹配的文档。相比由布尔模型检出的文档集，其主要效果在于，排序的文档提供了更精准的答案，更符合用户的信息需求。

定义 对于向量模型，项-文档对（k_i，d_j）的权重 $w_{i,j}$是非负的，而且是非二值的。假定所有的索引项是相互独立的，并表示成 t 维空间上的单位向量，其中 t 是索引项的总个数。文本 d_j 和查询 q 的表达形式是 t 维向量，如下所示。

$$\vec{d}_j = (w_{1,j}, w_{2,j}, \cdots, w_{t,j})$$

$$\vec{q} = (w_{1,q}, w_{2,q}, \cdots, w_{t,q})$$

77 其中，项-查询对（k_i，q）的权重是 $w_{i,q}$，$w_{i,q} \geqslant 0$。

因此，文档 d_j 和用户查询 q 被表示成 t 维向量，如图 3-9 所示。向量模型衡量文档 d_j 相对于查询 q 的相似度，即向量$\vec{d}_j$ 和向量$\vec{q}$之间的关联关系。这个关系可以被量化，例如用向量间的夹角余弦，也就是，

$$sim(d_j, q) = \frac{\vec{d}_j \cdot \vec{q}}{|\vec{d}_j| \times |\vec{q}|} = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}} \tag{3-10}$$

其中，$|\vec{d}_j|$和$|\vec{q}|$是文档和查询向量的范数，而$\vec{d}_j \cdot \vec{q}$是这两个向量间的内积。分母因数$|\vec{q}|$不影响排名（即文档的排序），因为它对所有的文档都一样。分母因数$|\vec{d}_j|$提供了文档长度的归一化。关于向量模型，更复杂的归一化可参考 Singhal、Buckley 和 Mitra 在主轴（pivoted）文档长度归一化上的工作 [1484]。

由于 $w_{i,j} \geqslant 0$ 且 $w_{i,q} \geqslant 0$，所以 $sim(q, d_j)$ 介于 0～1 之间。这样，向量模型根据文档相对于查询的相似度排序，而不是采用二值的判断标准。即使一篇文档和查询部分匹配，也可能被检出。例如，可以给 $sim(q, d_j)$ 设定一个阈值，然后检出相似度高于这个阈值的文档。

向量模型中的权重基本上是 TF-IDF 权重。这里，我们采用表 3-6 列出的权重框架 3，因此，

$$w_{i,q} = (1 + \log f_{i,q}) \times \log\left(\frac{N}{n_i}\right) \tag{3-11}$$

$$w_{i,j} = (1 + \log f_{i,j}) \times \log\left(\frac{N}{n_i}\right) \tag{3-12}$$

78 其中，$f_{i,q}$是索引项 k_i 在查询（或信息需求）q 中的频率。式（3-11）和式（3-12）应该仅用于项频大于零的值中。如果项频是零，那么相应的权重也是零。只要查询的项频都等于 1，这在 Web 中是常见的情况，权重 $w_{i,q}$可以简化为 IDF_i。在这种情况下，我们可把 $w_{i,j}$重定义为 $TF_{i,j}$。这样式（3-10）中分子的累加和是 $TF \times IDF$ 的形式，而不是 $TF \times IDF^2$。这是表 3-6 中权重框架 2 的解释。

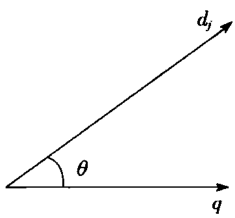


图 3-9 在向量模型中，采用 θ 的余弦作为 $sim(d_j, q)$

表 3-8 对于查询“to do”计算得到的文档排序，使用了式（3-11）和式（3-12）的 TF-IDF 权重（也可见图 3-2 和图 3-3）

文档	排序计算	排序
d_1	$\frac{1\times3+0.415\times0.830}{5.068}$	0.660
d_2	$\frac{1\times2+0.415\times0}{4.899}$	0.408
d_3	$\frac{1\times0+0.415\times1.073}{3.762}$	0.118
d_4	$\frac{1\times0+0.415\times1.073}{7.738}$	0.058

表 3-8 说明对于查询“to do”在图 3-6 的文档集中排序的计算。我们注意到文档 d_1 是排名最高的文档，因为它包含了所有查询项。而且，文档 d_3 和 d_4 仅含有查询项“do”，但文档 d_4 有一个较小的排序，因为它的向量范数更大（这是文档长度归一化的效果）。除了简单性外，由于向量模型采用的权重框架和文档长度归一化方法，因此它在一般文档集上获得了良好的结果。

向量模型主要的优点是：1）索引项权重公式提高了检索质量；2）它的部分匹配策略检出了近似于查询条件的文档；3）它的余弦排序公式根据文档相对于查询的相似度进行排序；4）文档长度归一化被自然地内建于排序中。理论上说，向量模型的缺点是索引项被假定为相互独立的（式（3-7）没有考虑索引项间的依赖性）。然而，实际上，利用项间依赖关系是有困难的。若处理得不恰当，甚至可能导致糟糕的结果。

除了简单性外，向量模型对一般文档集而言是一种弹性的排序策略。在没有查询扩展或相关反馈的情况下（见第 5 章），其产生的排序结果已很难进一步改进。其他大量排序方法已经和向量模型做了比较，但似乎对于一般文档集，向量模型是一个良好坚实并且简单快速的基本排序方法。由于这些原因，向量模型一直是一种流行的检索模型，在对替代排序公式和新提出的信息检索模型的评测中经常作为基准。

3.2.7 概率模型

概率模型于 1976 年由 Robertson 和 Sparck Jones [1365] 提出，这是一种基于概率框架的信息检索解决方案。其基本想法是，给定一个用户查询，有一个文档集恰好包含了所有相关的文档，且不包含其他文档。让我们把这个文档集称为理想答案集。给定这个理想答案集的描述，我们就可以检出相关的文档。因此，我们可以把查询过程看做是定义理想答案集属性的过程。问题是我们并不准确地知道这些属性是什么。我们所知道的是，有些索引项的语义可用来刻画这些属性。由于这些属性在查询时是未知的，因此最初要花费精力去估测这些属性是什么。这个最初的估测可使我们对理想答案集产生初步的概率描述，它可用来检出最初的文档集。 79

为了改善理想答案集的概率描述，可触发和用户间的互动。例如，用户可以检查检出的文档，并决定哪些文本是相关的，哪些是不相关的（在真实情况下，只有排在前面的文档需要被检查）。然后，系统可以用这些信息提炼理想答案的描述。通过多次重复这一过程，可以预期到理想答案的描述会变得更准确。因此，我们应该总是记住在最初需要估测理想答案集。而且，要有意识地用概率论的语言来建模。

概率模型的排序基于如下由 Robertson [1364] 阐述的基础假设。

定义 概率排序原则（Probability Ranking Principle）。*给定用户查询 q 和文档集中某篇文档 d_j，概率模型尝试估计用户认为文档 d_j 有趣（即相关）的概率。该模型假设相关概率仅仅取决于查询和文档的表示形式，即仅依赖系统可获得的信息。而且，该模型假设所有的文档中存在一个子集，该子集被用户看做是查询 q 的答案集。这个理想答案集被称为 R，它能最大化与用户相关的总体概率。集合 R 中的文档被预测为与查询 q 相关，不在这个集合中的文档被预测为不相关的。*

因为与用户的相关性可能被系统外的变量所影响，所以这一假设有些问题。用系统可获得的信息产生理想答案集，从用户的角度看可能并不理想（ideal）。而且，该原理没有清楚地说明如何计算相关概率。实际上，甚至没给出定义这种概率的样本空间。

给定查询 q，作为查询相似性的度量，概率模型给每篇文档赋一个比率 $P(d_j$ 与 q 相

关)/P(d_j 与 q 不相关)，它计算了文档 d_j 和查询 q 相关的优势比。按相关性的优势来排序使得错误判断的概率最小化 [599，1624]。

定义 在概率模型中，查询 q 是索引项的子集，文档 d_j 被表示成二值权重的向量，字体修改分量的值表明这个索引项是否出现，如下所示。

$$\vec{d}_j = (w_{1,j}, w_{2,j}, \cdots, w_{t,j})$$

其中，如果索引项 k_i 出现在文档 d_j 中，$w_{i,j}=1$；否则，$w_{i,j}=0$。给定查询 q，假设 R 是一个对于用户而言已知的相关文档集合（或初始猜测的）。假设 $\overline{R}$ 是 R 的补集（即不相关文档
80 集合），$P(R|\vec{d}_j, q)$ 是以 $\vec{d}_j$ 表示的文档 d_j 和查询 q 相关的概率。而且，$P(\overline{R}|\vec{d}_j, q)$ 是文档 d_j 和查询 q 不相关的概率。文档 d_j 和查询 q 的相似度 $sim(d_j, q)$ 定义为如下的比率。

$$sim(d_j, q) = \frac{P(R|\vec{d}_j, q)}{P(\overline{R}|\vec{d}_j, q)} \tag{3-13}$$

使用贝叶斯定理，

$$sim(d_j, q) = \frac{P(\vec{d}_j|R, q) \times P(R, q)}{P(\vec{d}_j|\overline{R}, q) \times P(\overline{R}, q)} = \frac{P(\vec{d}_j|R, q) \times P(R|q)}{P(\vec{d}_j|\overline{R}, q) \times P(\overline{R}|q)}$$

$P(\vec{d}_j|R, q)$ 表示从查询 q 的相关文档集 R 中随机选择的一篇文档表示为 $\vec{d}_j$ 的概率。而 $P(R|q)$ 表示从整个文档集中随机选择的文档和查询 q 相关的概率。$P(\vec{d}_j|\overline{R}, q)$ 和 $P(\overline{R}|q)$ 的含义是相似且互补的。

由于 $P(R|q)$ 和 $P(\overline{R}|q)$ 对集合内所有的文档都是相同的，因此可得，

$$sim(d_j, q) \sim \frac{P(\vec{d}_j|R, q)}{P(\vec{d}_j|\overline{R}, q)} \tag{3-14}$$

回顾一下，文档的表达式 $\vec{d}_j$ 是由二值权重组成，每一维表示索引项是否存在文档 d_j 中。如果假设索引项间的独立性，即所谓的二值独立假设（binary independence assumption），那么可得，

$$sim(d_j, q) \sim \frac{\left(\prod_{k_i|w_{i,j}=1} P(k_i|R, q)\right) \times \left(\prod_{k_i|w_{i,j}=0} P(\bar{k}_i|R, q)\right)}{\left(\prod_{k_i|w_{i,j}=1} P(k_i|\overline{R}, q)\right) \times \left(\prod_{k_i|w_{i,j}=0} P(\bar{k}_i|\overline{R}, q)\right)}$$

其中，$P(k_i|R, q)$ 表示索引项 k_i 出现在从查询 q 的相关文档集 R 中随机选择的一篇文档内的概率，而 $P(\bar{k}_i|R, q)$ 表示索引项 k_i 不出现在从查询 q 的相关文档集 R 中随机选择的一篇文档内的概率。集合 $\overline{R}$ 对应的概率有类似的含义。

为了简化表示，假设采用如下的约定，

$$p_{iR} = P(k_i|R, q)$$
$$q_{iR} = P(k_i|\overline{R}, q)$$

由于 $P(k_i|R, q)+P(\bar{k}_i|R, q)=1$，且 $P(k_i|\overline{R}, q)+P(\bar{k}_i|\overline{R}, q)=1$，因此可得，

81

$$sim(d_j, q) \sim \frac{\left(\prod_{k_i|w_{i,j}=1} p_{iR}\right) \times \left(\prod_{k_i|w_{i,j}=0} (1-p_{iR})\right)}{\left(\prod_{k_i|w_{i,j}=1} q_{iR}\right) \times \left(\prod_{k_i|w_{i,j}=0} (1-q_{iR})\right)}$$

使用对数函数不会改变排序而只会改变绝对的排序分数，因此可得，

$$sim(d_j,q) \sim \log \prod_{k_i|w_{i,j}=1} p_{iR} + \log \prod_{k_i|w_{i,j}=0} (1-p_{iR})$$

$$- \log \prod_{k_i|w_{i,j}=1} q_{iR} - \log \prod_{k_i|w_{i,j}=0} (1-q_{iR})$$

把能相互抵消的索引项累加起来，可得，

$$sim(d_j,q) \sim \log \prod_{k_i|w_{i,j}=1} p_{iR} + \log \prod_{k_i|w_{i,j}=0} (1-p_{iR})$$

$$- \log \prod_{k_i|w_{i,j}=1} (1-p_{iR}) + \log \prod_{k_i|w_{i,j}=1} (1-p_{iR})$$

$$- \log \prod_{k_i|w_{i,j}=1} q_{iR} - \log \prod_{k_i|w_{i,j}=0} (1-q_{iR})$$

$$+ \log \prod_{k_i|w_{i,j}=1} (1-q_{iR}) - \log \prod_{k_i|w_{i,j}=1} (1-q_{iR})$$

该式可以用对数函数重写为，

$$sim(d_j,q) \sim \log \prod_{k_i|w_{i,j}=1} \frac{p_{iR}}{(1-p_{iR})} + \log \prod_{k_i} (1-p_{iR})$$

$$+ \log \prod_{k_i|w_{i,j}=1} \frac{(1-q_{iR})}{q_{iR}} - \log \prod_{k_i} (1-q_{iR})$$

注意这里有两项是在所有索引项上的乘积，这意味着它们和查询 q 与文档 d_j 相独立。也就是说，它们是常数，对于排序是可以被忽略的。而且，假设

$$\forall k_i \notin q, \quad p_{iR} = q_{iR}$$

并把连乘的对数形式转变成对数的累加和形式，最终可得，

$$sim(d_j,q) \sim \sum_{k_i \in q \land k_i \in d_j} \log\left(\frac{p_{iR}}{1-p_{iR}}\right) + \log\left(\frac{1-q_{iR}}{q_{iR}}\right) \tag{3-15}$$

这是概率模型中计算排序的主要公式。

由于我们最初不知道集合 R，因此有必要设计一种方法用于最初计算概率 p_{iR} 和 q_{iR}。有许多方法可以进行这样的计算，我们下面对此加以讨论。

1. 索引项出现列联表

可以用如下所示的列联表［1365］来对概率排序公式给予更详细的解释。 82

定义　设 N 是文档集内文档的数量，n_i 是包含索引项 k_i 的文档的数量。而且，假设 R 是（从用户的角度看）和查询 q 相关的文档的总数量，r_i 是含有索引项 k_i 的相关文档的数量。索引项出现列联表如表 3-9 所示。

表 3-9　索引项出现列联表

情况	相关文档数	不相关文档数	总文档数
包含 k_i 的文档	r_i	n_i-r_i	n_i
不包含 k_i 的文档	$R-r_i$	$N-n_i-(R-r_i)$	$N-n_i$
所有文档	R	$N-R$	N

对于任何给定的查询，若都有索引项出现列联表中的信息（实际上我们不知道这信息，因为我们不知道新查询的相关文档），则可得，

$$p_{iR} = \frac{r_i}{R}, \quad q_{iR} = \frac{n_i - r_i}{N-R}$$

那么，式（3-15）可以重写为，

$$sim(d_j,q) \sim \sum_{k_i[q,d_j]} \log\left(\frac{r_i(N-n_i-R+r_i)}{(R-r_i)(n_i-r_i)}\right) \tag{3-16}$$

其中，$k_i[q, d_j]$ 是 $k_i \in q \wedge k_i \in d_j$ 的简短标记。为了让式（3-16）有效，我们仍然依赖于估计哪些是查询的相关文档。为了处理小值的 r_i，为方便起见在式（3-16）中给每一索引项增加 0.5 的数值，这样可得，

$$sim(d_j,q) \sim \sum_{k_i[q,d_j]} \log\left(\frac{(r_i+0.5)(N-n_i-R+r_i+0.5)}{(R-r_i+0.5)(n_i-r_i+0.5)}\right)$$

这个公式在极端情况下也能良好工作，在 r_i 和 R 是某些特定值的情况下也不会导致无限大的排序，例如 $R=r_i$。作为概率模型的经典排序等式，它也称为 Robertson-Sparck Jones 等式。

在没有 r_i 和 R 的估计值的情况下，上述等式无法计算。一种可能是假设 $R=r_i=0$，从而初始化排序公式可得，

$$sim(d_j,q) \sim \sum_{k_i \in q \wedge k_i \in d_j} \log\left(\frac{N-n_i+0.5}{n_i+0.5}\right) \tag{3-17}$$

在缺乏相关信息的情况下，这就是概率模型用于排序计算的公式。注意到，尽管存在一个 IDF 成分，但是既没有 TF 权重也没有文档长度归一化的支持。这些问题在 BM25 模型中
83 得以纠正，我们稍后讨论。

表 3-10 说明了对图 3-6 样例文档集查询“to do”的排序计算。在这种情况下，概率排序公式无法良好运作，因为索引项“do”产生的权重是负的。也就是，式（3-17）在 $n_i > N/2$ 的情况下无法正常工作，使得负的索引项被引入计算中。

一种避免这种反常情况的替代方式是从式（3-17）中去掉分子中的 n_i，如［1374，1371］中提出的。于是可得，

$$sim(d_j,q) \sim \sum_{k_i \in q \wedge k_i \in d_j} \log\left(\frac{N+0.5}{n_i+0.5}\right) \tag{3-18}$$

注意到现在出现在所有文档中的索引项（$n_i=N$）会产生等于零的权重，并且所有索引项的权重都是非负的。如表 3-11 所示，用式（3-18）为样例文档集重新计算了排序。现在这个概率排序和由向量模型生成的排序相似，即 $d_1 > d_2 > d_3 = d_4$。这里无法区别文档 d_3 和 d_4，因为与向量模型不同，它不支持文档长度归一化。

表 3-10 在图 3-6 中的样例文档集上，对查询“to do”用式（3-17）计算的文档排序

文档	排序计算	排序
d_1	$\log\frac{4-2+0.5}{2+0.5}+\log\frac{4-3+0.5}{3+0.5}$	-1.222
d_2	$\log\frac{4-2+0.5}{2+0.5}$	0
d_3	$\log\frac{4-3+0.5}{3+0.5}$	-1.222
d_4	$\log\frac{4-3+0.5}{3+0.5}$	-1.222

表 3-11 对于查询“to do”，用修正的概率式（3-18）计算的文档排序

文档	排序计算	排序
d_1	$\log\frac{4+0.5}{2+0.5}+\log\frac{4+0.5}{3+0.5}$	1.210
d_2	$\log\frac{4+0.5}{2+0.5}$	0.847
d_3	$\log\frac{4+0.5}{3+0.5}$	0.362
d_4	$\log\frac{4+0.5}{3+0.5}$	0.362

上面的例子考虑了 $r_i=R=0$ 的情况。另一种更仔细地估计参数 r_i 和 R 的方法是用式（3-18）做一次初始搜索，选择排序前 10～20 篇的文档，检查这些文档来获得对 r_i 和 R 新的估计，从集合中去掉这 10～20 篇文档，然后用估计获得的 r_i 和 R 重新检索查询——这一过程称为剩余排序［708］（residual ranking）。不幸的是，由于开始时需要人工干预来挑
84 选相关文档，因此这样的过程是不实用的。

2. 无相关信息下的排序

式（3-17）是概率模型在不提供相关信息的情况下计算排序的基本公式。如果要考虑相关信息，那么就需要人工干预。为了避免人工干预，Croft 和 Harper [452] 提出了一种修正参数 r_i 和 R 的自动过程。

为了在无相关信息的情况下计算式（3-15），我们可以：1）假设 p_{iR} 对所有索引项 k_i 是常数，典型值是 0.5，表示对这个项的分布没有先验知识；2）假设索引项在不相关文档中的分布可以通过索引项在文档集中所有文档上的分布估计。这两个假设产生了

$$p_{iR} = 0.5; \quad q_{iR} = \frac{n_i}{N}$$

代入式（3-15）中，可得：

$$sim(d_j, q) \sim \sum_{k_i \in q \wedge k_i \in d_j} \log\left(\frac{N - n_i}{n_i}\right) \tag{3-19}$$

这和式（3-17）相同，只是没有 0.5 的调整系数。给定这个初始猜测，我们能检出包含查询项的文档，并也能为这些文档提供一个初始的概率排序。接下来，我们可以尝试着改进这个初始排序，如下所示。

设 D 是由概率模型检出并排序得到的初始文档集的子集，例如可以定义为前 M 篇文档，其中 M 是事先定义的阈值。而且，设 D_i 是 D 的子集，由 D 中包含索引项 k_i 的文档组成。为了简单起见，我们也用 D 和 D_i 表示这些集合中元素的个数。为了提高排序质量，需要改进对 p_{iR} 和 q_{iR} 的猜测。这可以通过如下的假设获得：1）可以根据索引项 k_i 在当前检出文档中的分布来估计 p_{iR}；2）可以把所有未检出的文档看做是不相关的，以此来估计 q_{iR}。有了这些假设，可得：

$$p_{iR} = \frac{D_i}{D}; \quad q_{iR} = \frac{n_i - D_i}{N - D}$$

然后迭代重复这个过程。通过这样，我们期望在没有人工参与的情况下（这和最初的想法相反）改进对概率 p_{iR} 和 q_{iR} 的估测。当然，这完全取决于出现的索引项的模式，这些索引项是有噪声的（索引项本身的语义信息很少，可能会误导）。因此，当迭代次数增加时，这种方法不一定会提升结果。另一种方法是在定义子集 D 时就获得用户的协助（如最初构想的那样）。

p_{iR} 和 q_{iR} 最后的式子对于实际中产生的小值的 D 和 D_i 产生了问题，例如当 $D=1$ 和 $D_i=0$。为了绕开这些问题，可以增加调节因子，于是有：

$$p_{iR} = \frac{D_i + 0.5}{D + 1}; \quad q_{iR} = \frac{n_i - D_i + 0.5}{N - D + 1}$$ 85

调节因子设为常数 0.5 可能比较局限。一种方法是取分数 n_i/N 作为调节因子，于是有：

$$p_{iR} = \frac{D_i + \frac{n_i}{N}}{D + 1}; \quad q_{iR} = \frac{n_i - D_i + \frac{n_i}{N}}{N - D + 1}$$

这种情况下，排序计算是在不提供相关信息的情况下完全自动地进行的。

上述的排序公式对于图 3-6 中的样例文档集不适用，因为 N 的值太接近 n_i 了，以至于有些排序变成了负数。对于更大规模的实际文档集，这个式子是适用的。

3. 概率模型的优点和缺点

从理论上讲，概率模型的优点是它的最优性，即基于系统可获得的信息能够计算文档的相关概率，并按照降序排列。然而，由于文档的相关性受到系统之外因素的影响，因此这种

方法的实际效果并不好。其缺点包括：1）需要做初始估测把文档分为相关和不相关集合；2）这种方法实际上没有考虑到索引项出现在文档中的频率（即所有的权重是二值的）；3）缺乏文档长度归一化。概率模型的更高级变种，例如 BM-25 模型，修正了这些缺点，改进了检索效果。

3.2.8 经典模型之间的简单比较

布尔模型被认为是最弱的经典模型。其主要的问题是缺乏查询和文档之间的部分匹配，这经常导致糟糕的检索质量。对于概率模型是否比向量模型更好，尚有一些争议。Croft 进行了一些实验，表明概率模型能提供更好的检索质量［452］。然而，接下来由 Salton 和 Buckley 做的实验反驳了这种观点［1410］。通过多个不同的评测指标，Salton 和 Buckley 证实了对于一般的文档集，向量模型超过了概率模型。这似乎也是信息检索领域的研究人员和从业人员主要的观点。

对于一般的文档集，向量模型提供了一个合理健壮的信息检索模型供比较。为了更好地理解这点，可以做如下的尝试。找一个含有上万篇文档的集合作为参考集。构想一种新的排序公式，与我们到现在为止所讨论的都不相同。接下来，实现经典向量模型。测试并比较由新的排序模型和向量模型产生的排序。这可以通过在文档集上运行一组预选的查询，比较两种排序公式产生的答案来实现（将在第 4 章讨论如何做对比评价）。尝试着改进新的排序公式，使其结果和向量模型的结果接近。很有趣的是，这不简单，也不容易做。向量模型的权
86 重公式植根于信息论，提供了一种对一般文档集而言简单但有效的排序公式。

3.3 其他集合论模型

本节讨论其他三种集合论模型，即基于集合的模型、扩展布尔模型和模糊集模型。

3.3.1 基于集合的模型

基于集合的模型是一种较新的方法，结合了集合论与向量空间模型的排序。尽管它同时有布尔模型和代数模型的特征，但我们还是把它看做是布尔模型。基于集合的模型的基本思想是采用索引项之间的相互依赖性来提高检索结果。索引项之间的依赖信息是通过引入项集（termset）来获得的，这些项集是相互关联的索引项的集合。该方法在各种文档集上都获得了较好的检索结果。作为首个有效地采用了索引项依赖性的方法，它改进了一般文档集的检索结果，并维持了低廉的计算成本。我们的讨论受了［1295］很大影响。

1. 项集

基于集合的模型中的主要概念是项集，它是用来替换索引项的。也就是说，这个模型不是采用标准索引项作为基本分量，而是采用项集对文档和查询建立索引。让我们把项集的概念规范化。

定义 *项集 $S_i=\{k_a, k_b, \cdots, k_n\}$ 是文档集中索引项的子集。若 S_i 中所有的索引项都出现在文档 d_j 中，那么我们说项集 S_i 出现在 d_j 中。设 N_i 是包含 S_i 的文档的数量。*

注意，在文档集合中有 2^t 个项集。然而，数据集中实际的项集数量远小于 2^t，因为大部分索引项的组合没有语义含义。

定义 *如前所述，设 t 是文档集中索引项的数量。那么，由文档集内所有项集组成的集合 $V_S=\{S_1, S_2, \cdots, S_{2^t}\}$ 是文档集的词汇表集合。*

举例来说，考虑图 3-6 所示的文档集。为了简化记号，我们定义

$a = k_a =$ to　　$f = k_f =$ not　　$k = k_k =$ therefore
$b = k_b =$ do　　$g = k_g =$ I　　$l = k_l =$ da
$c = k_c =$ is　　$h = k_h =$ am　　$m = k_m =$ let
$d = k_d =$ be　　$i = k_i =$ what　　$n = k_n =$ it
$e = k_e =$ or　　$j = k_j =$ think

图 3-10 说明了采用这种记号约定，文档集所具有的一种更简单的表示形式。假设用户定义了查询 q 是“to do be it”（或者 $q=\{a, b, d, n\}$）。对于这个查询，词汇表集合如表 3-12 所示。我们观察到，对于由 q 形成的最多 15 个项集，在文档集中有 11 个，例如 $S_a=\{a\}$ 和 $S_{abd}=\{a, b, d\}$。 87

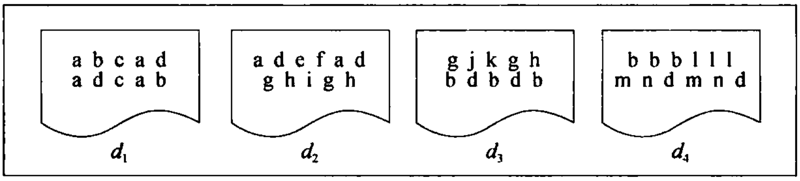


图 3-10　图 3-6 中的文档集，现在用字母表示索引项

表 3-12　查询 $q=\{a, b, d, n\}$ 的词汇表集合

项集	项的集合	文档	项集	项的集合	文档
S_a	$\{a\}$	$\{d_1, d_2\}$	S_{bd}	$\{b, d\}$	$\{d_1, d_3, d_4\}$
S_b	$\{b\}$	$\{d_1, d_3, d_4\}$	S_{bn}	$\{b, n\}$	$\{d_4\}$
S_d	$\{d\}$	$\{d_1, d_2, d_3, d_4\}$	S_{dn}	$\{d, n\}$	$\{d_4\}$
S_n	$\{n\}$	$\{d_4\}$	S_{abd}	$\{a, b, d\}$	$\{d_1\}$
S_{ab}	$\{a, b\}$	$\{d_1\}$	S_{bdn}	$\{b, d, n\}$	$\{d_4\}$
S_{ad}	$\{a, d\}$	$\{d_1, d_2\}$			

一项重要的观察是，基于集合的模型需要计算的只是查询的项集。因此，在查询处理时，只需要考虑由查询生成的项集。对于短查询，这是相当有效的。

对于长查询，可能需要计算和考虑更多的项集。一种减少项集数量的方法是仅仅考虑那些在文档集中高于某个最小出现频率的项集。我们把这一想法按如下方式规范化。

定义　由 n 个项构成的项集称为 n 项集。如果包含 n 项集的文档数量 $\mathcal{N}_i$ 高于某个给定的阈值，那么这个 n 项集 S_i 称为是频繁的。这意味着 n 项集是频繁的，当且仅当它所包含的全部 $(n-1)$ 项集也是频繁的。

后面的现象最初在 [21] 中阐述，这加快了与每个项集关联的文档集的计算（即项集倒排表）。举例来说，重新考察图 3-10 中的文档集。设项集的频率阈值是 2。为了计算查询 $q=\{a, b, d, n\}$ 所有的频繁项集，按如下步骤计算更大的项集直到找不到项集为止。

1）计算频繁的 1 项集及其文档倒排表。

- $S_a \rightarrow \{d_1, d_2\}$ 88
- $S_b \rightarrow \{d_1, d_3, d_4\}$
- $S_d \rightarrow \{d_1, d_2, d_3, d_4\}$

2）组合这些倒排表来计算频繁 2 项集。

- $S_{ad} \rightarrow \{d_1, d_2\}$
- $S_{bd} \rightarrow \{d_1, d_3, d_4\}$

3）组合上述倒排表，发现没有频繁 3 项集，停止。

我们注意到在文档集中只有 5 个频繁项集，并且没有频繁项集的规模是大于或者等于 3 的。

最重要的，这个例子说明我们能高效地从频繁 1 项集的倒排表计算出频繁 n 项集。因此，所需的索引结构仅是标准倒排索引（见第 9 章）。对于包含最多 4～5 个项的查询，这是相当快的。

2. 排序计算

排序计算基于向量模型，主要的区别是向量模型是由项集形成的，而不是由索引项形成的。对此，我们给项集一个权重，如下所示。

定义 给定查询 q，由索引项集合产生，设 $\{S_1, S_2, \cdots\}$ 是由 q 产生的所有项集的集合。如前所述，设 $\mathcal{N}_i$ 是包含项集 S_i 的文档数量，并设 N 是文档集中的总文档数量。定义 $\mathcal{F}_{i,j}$ 是项集 S_i 在文档 d_j 中的原始出现频率。对于 (S_i, d_j)，赋予项集权重 $\mathcal{W}_{i,j}$

$$\mathcal{W}_{i,j} = \begin{cases} (1+\log \mathcal{F}_{i,j})\log\left(1+\dfrac{N}{\mathcal{N}_i}\right) & \mathcal{F}_{i,j} > 0 \\ 0 & \mathcal{F}_{i,j} = 0 \end{cases} \tag{3-20}$$

这是 TF-IDF 权重的变体。而且，对于 (S_i, q) 有类似的计算，并给予项集权重 $\mathcal{W}_{i,q}$。

举例来说，再次考虑查询 $q=\{a, b, d, n\}$，文档 d_1，并假设频率的最小阈值是 1。关注的权重值如表 3-13 所示。

表 3-13 基于集合的模型中的权重

项集	权重	log TF	IDF	TF-IDF
S_a	$\mathcal{W}_{a,1}$	1+log 4	log(1+4/2)	4.75
S_b	$\mathcal{W}_{b,1}$	1+log 2	log(1+4/3)	2.44
S_d	$\mathcal{W}_{d,1}$	1+log 2	log(1+4/4)	2.00
S_n	$\mathcal{W}_{n,1}$	0	log(1+4/1)	0.00
S_{ab}	$\mathcal{W}_{ab,1}$	1+log 2	log(1+4/1)	4.64
S_{ad}	$\mathcal{W}_{ad,1}$	1+log 2	log(1+4/2)	3.17
S_{bd}	$\mathcal{W}_{bd,1}$	1+log 2	log(1+4/3)	2.44
S_{bn}	$\mathcal{W}_{bn,1}$	0	log(1+4/1)	0.00
S_{dn}	$\mathcal{W}_{dn,1}$	0	log(1+4/1)	0.00
S_{abd}	$\mathcal{W}_{abd,1}$	1+log 2	log(1+4/1)	4.64
S_{bdn}	$\mathcal{W}_{bdn,1}$	0	log(1+4/1)	0.00

定义 在基于集合的模型中，文档 d_j 和查询 q 被表示成 2^t 维项集空间中的向量，如下所示。

$$\vec{d_j} = (\mathcal{W}_{1,j}, \mathcal{W}_{2,j}, \cdots, \mathcal{W}_{2^t,j})$$

$$\vec{q} = (\mathcal{W}_{1,q}, \mathcal{W}_{2,q}, \cdots, \mathcal{W}_{2^t,q})$$

其中项集权重如前定义。文档 d_j 相对于查询 q 的排序（或者相似度）定义为文档向量和查询向量之间的夹角余弦，如下所示。

89

$$sim(d_j, q) = \frac{\vec{d_j} \cdot \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{S_i} \mathcal{W}_{i,j} \times \mathcal{W}_{i,q}}{|\vec{d_j}| \times |\vec{q}|} \tag{3-21}$$

对于不是从查询 q 中生成的项集，$\mathcal{W}_{i,q}=0$。也就是说，基于集合的模型把排序计算限制在由查询生成的项集上。

文档范数 $|\vec{d_j}|$ 在项集空间上难于计算。因此，出于效率的考虑，其计算限制在 1 项集，即文档范数在索引项的空间中计算，就像向量模型那样。除了获得快速的计算外，基于 1 项集的文档范数对长文档不利，这是文档归一化的主要目的。

再次考察查询 $q=\{a, b, d, n\}$ 和文档 d_1。为了计算文档 d_1 的排序，我们需要考察 7 个权重为非零的项集：S_a、S_b、S_d、S_{ab}、S_{ad}、S_{bd} 和 S_{abd}。而且，我们需要计算 1 项集的文档范数，如下所示。

$$\begin{aligned} |\vec{d}_1| &= \sqrt{\mathcal{W}_{a,1}^2 + \mathcal{W}_{b,1}^2 + \mathcal{W}_{c,1}^2 + \mathcal{W}_{d,1}^2} \\ &= \sqrt{4.75^2 + 2.44^2 + 4.64^2 + 2.00^2} \\ &= 7.35 \end{aligned} \tag{3-22}$$

其中，$\mathcal{W}_{c,1}=(1+\log 2)\log(1+4/1)=4.64$。因为查询对所有文档是相同的，因此其范数不会影响排序，不需要包含在排序计算中（与向量模型中的情况一样）。而且，查询项权重由各自的 IDF 权重给定，因为这种情况下的查询项频都是 1（在 Web 中这也是很常见的情况）。因此，文档 d_1 的排序可按如下计算：

$$\begin{aligned} sim(d_1,q) &= (\mathcal{W}_{a,1}\times\mathcal{W}_{a,q}+\mathcal{W}_{b,1}\times\mathcal{W}_{b,q}+\mathcal{W}_{d,1}\times\mathcal{W}_{d,q}+\mathcal{W}_{ab,1}\times\mathcal{W}_{ab,q}+ \\ &\quad \mathcal{W}_{ad,1}\times\mathcal{W}_{ad,q}+\mathcal{W}_{bd,1}\times\mathcal{W}_{bd,q}+\mathcal{W}_{abd,1}\times\mathcal{W}_{abd,q})/|\vec{d}_1| \\ &= (4.75\times1.58+2.44\times1.22+2.00\times1.00+4.64\times2.32+ \\ &\quad 3.17\times1.58+2.44\times1.22+4.64\times2.32)/7.35 \\ &= 5.71 \end{aligned}$$

90

我们注意到有 7 个项集和 d_1 的排序有关，而不仅仅是 3 个权重为非零的 1 项集（对于向量模型来说就是如此）。

3. 使用闭项集的更快速排序

频繁项集的概念使得排序计算限制在那些在文档集中出现次数高于最小阈值的项集上。然而，在大的文档集中有许多频繁项集。这是个问题，因为长查询需要考虑的项集的数量是非常多的。

一种可能的方法是把排序计算进一步限制在一组更小数目的项集上，这可以通过观察项集的一些诸如闭包之类的属性来获得。对此，我们查看项集之间的关系，以及它们所出现的文档之间的关系。例如，在表 3-12 中我们观察到

$$S_a \subset S_{ab}$$
$$S_n \subset S_{bn}$$

然而，虽然 S_n 和 S_{bn} 出现在相同的文档集合中时，S_a 和 S_{ab} 却出现在不同的文档集合中。这直接影响了项集的闭包，如下所示。

定义 *在文档集 C 中，项集 S_i 的闭包是所有和 S_i 在同一个文档子集中共现的频繁项集的集合。给定 S_i 的闭包，其中最大的项集被称为闭项集，记做 S_{Φ_i}。按如下的方法进行规范化。*

- *设 $D_i \subseteq C$ 是频繁项集 S_i 出现过的所有文档的子集。*
- *设 $S(D_i)$ 是由所有出现且仅出现在 D_i 中的频繁项集组成的集合。这是 S_i 的闭包。*

那么，闭项集 S_{Φ_i}，满足如下的性质

$$\nexists S_j \in S(D_i) \mid S_{\Phi_i} \subset S_j$$

举例来说，表 3-14 显示了我们样例文档集中所有的频繁项集和闭项集，其中假设最小阈值等于 2。

表 3-14 文档集上的频繁项集和闭项集，最小频率阈值等于 2

频度（S_i）	频繁项集	闭项集
4	d	d
3	b，bd	bd
2	a，ad	ad
2	g，h，gh，ghd	ghd

闭项集是有趣的，因为它们包括了出现在相同文档集合内的较小的项集。因此，当我们计算排序时使用一个闭项集，那些已经

91 被囊括在内的频繁项集就不必包含在内了。例如，如果我们计算文档 d_1 相对于查询 $q=\{a, b, d, n\}$ 的排序，其中仅考虑高于最小频率阈值为 2 的频繁项集和闭项集，可得：

$$\begin{aligned} sim(d_1,q) &= (\mathcal{W}_{d,1}\times\mathcal{W}_{d,q}+\mathcal{W}_{ad,1}\times\mathcal{W}_{ad,q}+\mathcal{W}_{bd,1}\times\mathcal{W}_{bd,q})/|\vec{d}_1| \\ &= (2.00\times1.00+3.17\times1.58+2.44\times1.22)/7.35 \\ &= 1.35 \end{aligned}$$

我们注意到现在有 5 个项集影响 d_1 的排序，而不是像前面那样有 7 个。我们也注意到 d_1 的排序减小了，所有含有这个闭项集的文档的排序也都是这样，因为闭项集（根据定义）会影响一组相同的文档集合。因此，如果把排序计算限制在闭项集内，那么能期望减少查询处理时间。闭项集数量越少，查询处理时间也减少得越多。

除了闭项集外，还有最大项集。这些可以用来自动地从查询构建出一组子查询。例如，在 Web 中，这种查询构建可能会带来更好的检索结果，如 [1297] 所讨论的那样。

3.3.2 扩展布尔模型

布尔检索简单优雅。然而，由于它不支持索引项权重，因此它也不生成答案集的排序。因此，输出的规模可能过大或过小。由于这些问题，现代信息检索系统不再基于布尔模型。实际上，大部分新系统其核心采用某种形式的向量检索。其原因是向量空间模型简单、快速，能产生更好的检索质量。另一种方法是用部分匹配和项权重的功能来扩展布尔模型。这种方法使得人们可以把布尔查询表达式和向量模型的特点结合起来。接下来，我们要讨论的是众多模型中的一种，它用向量模型的特征来扩展布尔模型。

扩展布尔模型，由 Salton、Fox 和 Wu [1412] 于 1983 年引入，基于对布尔逻辑基本假设的如下反思。考察一个合取布尔查询 $q=k_x\wedge k_y$。根据布尔模型，一篇仅包含 k_x 或者 k_y 其中之一的文档和另一篇不包含其中任何一个的文档都是不相关的。然而，这种二值决策准则通常与常识不相符。当人们考察纯粹的析取查询时，同理可得类似的情况。

当只考察两个索引项时，我们把查询和文档在二维图中绘制出来，如图 3-11 所示。文档 d_j 在这个空间中通过采用与二元组（k_x，d_j）和（k_y，d_j）伴随的权重 $w_{x,j}$ 和 $w_{y,j}$ 来定位。假设这些权重被归一化，并置于 0～1 之间。举例来说，这些权重可以算成是如下的归
92 一化 TF-IDF 因子。

$$w_{x,j}=\frac{f_{x,j}}{\max_x f_{x,j}}\times\frac{IDF_x}{\max_i IDF_i} \tag{3-23}$$

其中 $f_{x,j}$ 是索引项 k_x 在文档 d_j 中的项频，IDF_i 是索引项 k_i 的反比文档频率。出于简单考虑，在本节剩下来的部分，我们称权重 $w_{x,j}$ 为 x，权重 $w_{y,j}$ 为 y，文档向量 $\vec{d}_j=(w_{x,j}, w_{y,j})$ 为点 $d_j=(x, y)$。

观察图 3-11，我们注意到两个特殊的地方。首先，对于析取查询 $q_{or}=k_x\vee k_y$，点（0，0）是最无须关注的点。这表明可以把到（0，0）距离作为对于查询 q_{or} 相似度的度量。其次，对于合取查询 $q_{and}=k_x\wedge k_y$，点（1，1）是我们最关注的点。这表明可以把到点（1，1）距离的补作为对于查询 q_{and} 相似度的度量。接着，归一化这些距离，可得：

$$sim(q_{or},d)=\sqrt{\frac{x^2+y^2}{2}}$$

$$sim(q_{and},d)=1-\sqrt{\frac{(1-x)^2+(1-y)^2}{2}} \tag{3-24}$$

如果权重都是布尔值（即 $w_{x,j}\in\{0,1\}$），那么文档总是定位在四角中的一角（即（0，0）、（0，1）、（1，0）或者（1，1）），并且 $sim(q_{or}, d)$ 的值被限定在 0、$1/\sqrt{2}$和 1。类似地，$sim(q_{and}, d)$ 的值被限定在 0、$1-1/\sqrt{2}$和 1。然而，通常情况下权重是采用非布尔值的。

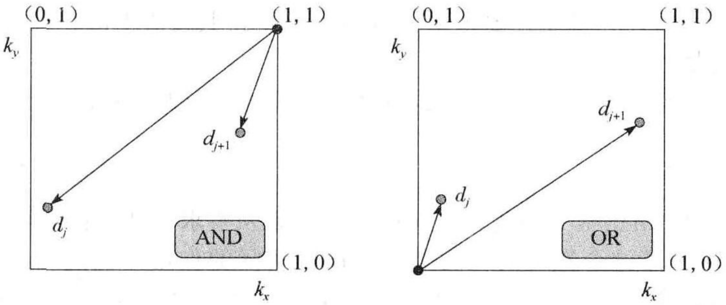


图 3-11　扩展布尔逻辑，考察仅由两个索引项 k_x 和 k_y 构成的空间

给定文档集中的索引项的个数 t，上面讨论的布尔模型可以自然地拓展到考察 t 维空间上的欧几里得距离。然而，一种更全面的泛化形式是如下的采用向量范数的理论。

p 范数模型（p-norm model）泛化了距离的概念，不仅包括欧几里得距离，也包括了 p 距离，其中 $1\leqslant p\leqslant\infty$是新引入的参数，它的值必须在查询时定义。那么，泛化的析取查询 93
可以表示为

$$q_{or}=k_1 \vee^p k_2 \vee^p \cdots \vee^p k_m$$

类似地，泛化的合取查询可以表示为

$$q_{and}=k_1 \wedge^p k_2 \wedge^p \cdots \wedge^p k_m$$

对应的查询-文档相似度现在可以表示为

$$sim(q_{or}, d_j)=\left(\frac{x_1^p+x_2^p+\cdots+x_m^p}{m}\right)^{\frac{1}{p}}$$

$$sim(q_{and}, d_j)=1-\left(\frac{(1-x_1)^p+(1-x_2)^p+\cdots+(1-x_m)^p}{m}\right)^{\frac{1}{p}} \tag{3-25}$$

其中，每个 x_i 表示二元组（k_i，d_j）对应的权重 $w_{i,j}$。

如上定义的 p 范数拥有一些的有趣性质，如下所示。首先，当 $p=1$ 时，能够验证

$$sim(q_{or}, d_j)=sim(q_{and}, d_j)=\frac{x_1+\cdots+x_m}{m}$$

其次，当 $p=\infty$时，可以验证

$$sim(q_{or}, d_j)=\max(x_i)$$

$$sim(q_{and}, d_j)=\min(x_i)$$

因此，当 $p=1$ 时，合取与析取查询是以项-文档权重的累加形式来评估的，就像向量模型的排序公式（计算内积）一样。而且，当 $p=\infty$时，查询是根据模糊逻辑（可看做布尔逻辑的泛化）的形式来评估的。把参数 p 在 $1\sim\infty$之间变动，我们能把 p 范数的排序行为从向量模型式的排序改为布尔模型式的排序。这是相当有用的，并且是支持扩展布尔模型的一种好论据。

对于更一般查询的处理是通过把操作符按照预先定义的方式聚合起来。举例来说，考察查询 $q=(k_1\wedge^p k_2)\vee^p k_3$。文档 d_j 和查询之间的相似度 $sim(q, d_j)$ 可以如下计算得到：

$$sim(q,d) = \left[\frac{\left(1-\left(\frac{(1-x_1)^p+(1-x_2)^p}{2}\right)^{\frac{1}{p}}\right)^p + x_3^p}{2}\right]^{\frac{1}{p}} \qquad (3\text{-}26)$$

不管与/或操作符的数量是多少，这个过程都可以递归执行。然而，如果我们把析取式展开为合取式，并重新计算排序，那么我们得到的排序和上述值不同。也就是说，逻辑布尔
94 操作符不保序，这是扩展布尔模型的缺点。

扩展布尔模型的另一个有趣的方面是，可以在相同的查询需求中采用不同的 p 值组合。例如，查询

$$(k_1 \lor^2 k_2) \land^\infty k_3$$

可以用来表示 k_1 和 k_2 用在基于向量的系统中，但是必须要求 k_3 出现（即合取操作可看做是布尔操作）。尽管不清楚这个额外的功能是否有任何实际的作用，但这个模型允许了这个功能，并以自然的方式实现了这个功能（不需要笨拙的扩展方法来对付特殊情况）。

我们观察到扩展布尔模型扩展了布尔代数，用代数距离来解释布尔操作符。在这个意义上，这是真正的同时包含了集合论模型和代数模型性质的混合模型。为了简单起见，我们选择把该模型归类为集合论模型中。

3.3.3 模糊集模型

把文档和查询表示为关键词的集合，所产生的描述，仅仅与对应的文档和查询的真实语义内容部分相关。文档对于查询项的匹配是近似的（或模糊的）。这可以通过如下方式建模，对于每个查询项定义一个模糊集，每篇文档在这个集合中都有某种程度的隶属度（通常小于1）。用模糊论的概念来解释检索过程是过去几年中提出的各种不同的模糊集信息检索模型的基础。这里我们不去回顾那么多模型，而是集中在一个特殊的模型，其描述和本章中已经描述的模型匹配得很好。我们的讨论是基于由 Ogawa、Morita 和 Kobayashi [1224] 提出的用于信息检索的模糊集模型。在这之前，我们快速地介绍一些基本概念。

1. 模糊集理论

模糊集理论 [1763] 处理边界不明确的类的表示。主要的思想是给类中的每个元素赋予一个隶属函数。该函数的值介于区间 [0，1]，其中 0 表示没有隶属关系，1 对应着完全隶属关系。介于 0～1 之间的隶属关系值表示了类别中的边际元素。因此，模糊集中的隶属关系是一种渐变的，而不是像传统的布尔逻辑中那样突变的概念。

定义 论域 U（universe of discourse）的模糊子集 A 由隶属函数 μ_A：$U \rightarrow [0，1]$ 来表示，该函数给 U 中每个元素 u 赋予介于区间 [0，1] 的值 $\mu_A(u)$。

模糊集最常采用的三项操作是：模糊集的补，两个或多个模糊集的并，两个或多个模糊
95 集的交。其定义如下所示。

定义 设 U 是论域，A 和 B 是 U 的两个模糊子集，并且 $\overline{A}$ 是 A 相对于 U 的补集。此外，假设 u 是 U 的一个元素。那么，

$$\mu_{\overline{A}}(u) = 1 - \mu_A(u) \qquad (3\text{-}27)$$

$$\mu_{A\cup B}(u) = \max(\mu_A(u), \mu_B(u)) \qquad (3\text{-}28)$$

$$\mu_{A\cap B}(u) = \min(\mu_A(u), \mu_B(u)) \qquad (3\text{-}29)$$

用模糊集来表示模糊性和不准确性是有价值的，并已经被应用到不同的领域中。下面我们将讨论它们在信息检索中的应用。

2. 模糊信息检索

另一种对信息检索过程建模的方法是使用同义词典，其中定义了索引项间的关系。基本

的思想是用来自词典的相关项扩展查询的索引项，使得额外的相关文档（即除了常规检索得到的文档外）也能被用户的查询检索到。同义词典也能用来对模糊集的信息检索问题建模，如下所示。

同义词典可以根据项间相关性矩阵 **C** 来构建（在［1224］中称为关键词连接矩阵），其行和列是与文档集中的索引项相关联的。这正是 3.2.3 节中定义的项间相关性矩阵，但其相关性权重不同。在这种情况下，矩阵 **C**、索引项 k_i 和 k_l 之间的归一化相关因子 $c_{i,l}$ 可定义为

$$c_{i,l} = \frac{n_{i,l}}{n_i + n_l - n_{i,l}} \tag{3-30}$$

其中 n_i 是含有索引项 k_i 的文档数量，n_l 是含有索引项 k_l 的文档数量，$n_{i,l}$ 是同时含有这两个索引项的文档数量。这样的相关性度量是相当常见的，并广泛应用于聚类算法中，第 8 章会加以详细讨论。

我们用项间相关性矩阵 **C** 对每一个索引项 k_i 赋予一个模糊集。在这个模糊集中，文档 d_j 有如下定义的归属关系 $\mu_{i,j}$：

$$\mu_{i,j} = 1 - \prod_{k_l \in d_j} (1 - c_{i,l}) \tag{3-31}$$

它计算了文档 d_j 上所有索引项的代数和（这里实现为 $c_{i,l}$ 补的代数积的补）。若文档 d_j 自己的索引项和索引项 k_i 相关，那么该文档就隶属于索引项 k_i 的模糊集。只要 d_j 中至少有一个索引项 k_l 和索引项 k_i 有紧密的联系（即 $c_{i,l}\sim1$），那么 $\mu_{i,j}\sim1$，并且索引项 k_i 是文档 d_j 的一个良好的模糊索引项。如果 d_j 所有的索引项都仅和 k_i 松散地关联，也就是说 $\mu_{i,j}\sim0$，那么索引项 k_i 就不是 d_j 的良好模糊索引项。采用文档 d_j 上所有索引项的代数和（而不是标准的 max 函数）使得 $\mu_{i,j}$ 的值变得平滑。 96

用户通过提供类似布尔表达式的查询表达式来表明信息需求。像经典布尔模型一样，这个查询被转换为析取范式的形式。例如，查询［$q=k_a \wedge (k_b \vee \neg k_c)$］可以写成［$\vec{q}_{dnf}=(1,1,1)\vee(1,1,0)\vee(1,0,0)$］的析取范式形式，其中每一个分量是和三元组（$k_a$，$k_b$，$k_c$）相关联的二值带权向量。这些二值带权向量是 $\vec{q}_{dnf}$ 的合取分量。设 cc_i 表示第 i 个合取分量。从总体上看，

$$\vec{q}_{dnf} = cc_1 \vee cc_2 \vee \cdots \vee cc_p$$

其中 p 是 $\vec{q}_{dnf}$ 合取分量的个数。计算文档与查询相关度的过程类似于经典布尔模型采用的过程。区别是我们这里要处理模糊集（而不是明确集或者布尔集）。我们接下来看一个例子。

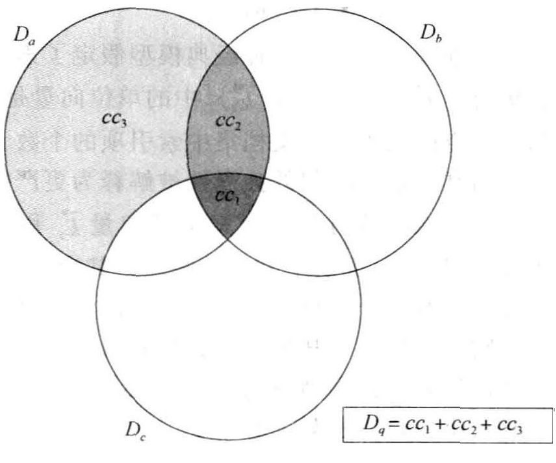


图 3-12　查询［$q=k_a \wedge (k_b \vee \neg k_c)$］的模糊集，其中每个分量 cc_i，$i\in\{1, 2, 3\}$ 是合取分量。D_q 是查询的模糊集

重新考察查询［$q=k_a \wedge (k_b \vee \neg k_c)$］。设 D_a 是和索引项 k_a 相关联的文档的模糊集。这个集合可以是由隶属度 $\mu_{a,j}$ 大于某个预定阈值 K 的文档组成的。而且，设 $\overline{D}_a$ 是集合 D_a 的补集。模糊集 $\overline{D}_a$ 对应于索引项 $\overline{k}_a$，$\overline{k}_a$ 是索引项 k_a 的非操作。类似地，我们可以分别定义与索引项 k_b 和 k_c 对应的模糊集 D_b 和 D_c。图 3-12 说明了这个例子。由于集合都是模糊的，因此即使文档 d_j 中没有提到索引项

k_a，但文档 d_j 仍有可能属于集合 D_a。

图 3-12 中的查询模糊集 D_q 是 $\vec{q}_{dnf}$ 的三个合取分量对应的模糊集的并，这三个分量这里是 cc_1、cc_2 和 cc_3。在这种情况下，我们有 $cc_1=\mu_{a,j}\mu_{b,j}\mu_{c,j}$、$cc_2=\mu_{a,j}\mu_{b,j}(1-\mu_{c,j})$ 和 $cc_3=\mu_{a,j}(1-\mu_{b,j})(1-\mu_{c,j})$，其中 $\mu_{a,j}$、$\mu_{b,j}$ 和 $\mu_{c,j}$ 是文档 d_j 分别对于模糊集 D_a、D_b 和 D_c 的隶属
97 度。文档 d_j 在模糊集 D_q 的隶属度 $\mu_{q,j}$ 可以如下计算得到：

$$\begin{aligned}\mu_{q,j} &= \mu_{\alpha_1+\alpha_2+\alpha_3,j}\\ &= 1-\prod_{i=1}^{3}(1-\mu_{\alpha_{i,j}})\\ &= 1-(1-\mu_{a,j}\mu_{b,j}\mu_{c,j})\times(1-\mu_{a,j}\mu_{b,j}(1-\mu_{c,j}))\times(1-\mu_{a,j}(1-\mu_{b,j})(1-\mu_{c,j}))\end{aligned}$$

正如已经看到的，析取模糊集的隶属度是由代数和计算得到的，而不是更常见的 max 函数。而且，合取模糊集的隶属度是由代数积得到的，而不是更常见的 min 函数。用代数和与代数积产生的隶属度比那些用 min 和 max 函数算得的情况变化得更平滑，这似乎更适合信息检索系统。

我们的例子说明了模糊模型是如何对与用户查询相关的文档排序的。这个模型用一个项间相关性矩阵计算文档 d_j 及其模糊索引项之间的关系。此外，这个模型采用代数和与代数积（而不是 max 和 min）来计算文档 d_j 在用户查询定义的模糊集中的总体隶属度。Ogawa、Morita 和 Kobayashi [1224] 也讨论了如何把用户的反馈信息整合到模型中，但是这样的讨论超越了本章的范围。

用于信息检索的模糊集模型主要在模糊理论的文献中讨论，在信息检索领域并不流行。此外，模糊集模型的大多数实验仅考察了小规模文档集，难于同时进行比较。除此之外，它们对信息检索问题的建模提供了有趣的框架，自然地纳入了索引项相关性。

3.4 其他代数模型

本节讨论三种其他的代数模型，也就是广义向量空间模型（generalized vector space model）、潜在语义索引模型（latent semantic indexing model）和神经网络模型（neural network model）。

3.4.1 广义向量空间模型

如前面讨论的，三种经典模型假定了索引项之间的独立性。对于向量模型，该假设意味着集合 $\{\vec{k}_1, \vec{k}_2, \cdots, \vec{k}_t\}$ 中的单位向量是线性独立的，并形成了我们所关注的子空间的基。该空间的维度是文档集中索引项的个数 t。

通常，索引项间的独立性被解释为更严格的意义，即用索引项向量间两两正交的性质来表示，这意味着对每一对索引项向量 $\vec{k}_i$ 和 $\vec{k}_j$，有 $\vec{k}_i \cdot \vec{k}_j=0$。1985 年，Wong、Ziarko 和 Wong [1718] 提出了另一种解释，其中索引项不是两两正交的。它们本身是由更小的成分
98 构成，这些成分来自手边的特殊文档集。这样的解释导致了广义向量空间模型。

定义 给定文档集中索引项的词汇表 $V=\{k_1, k_2, \cdots, k_t\}$，像经典布尔模型那样，考察从词汇表中生成的所有 2^t 个索引项合取分量。每个合取分量称为最小项 m_r(minterm)。所有最小项的列表如下所示。

$$\begin{aligned} &(k_1,k_2,k_3,\cdots,k_t)\\ m_1&=(0\ ,0\ ,0\ ,\cdots,0)\\ m_2&=(1\ ,0\ ,0\ ,\cdots,0)\end{aligned}$$

$$m_3 = (0\ ,1\ ,0\ ,\cdots,0)$$
$$m_4 = (1\ ,1\ ,0\ ,\cdots,0)$$
$$\vdots$$
$$m_{2^t} = (1\ ,1\ ,1\ ,\cdots,1)$$

定义

$$on(i,m_r) = \begin{cases} 1 & k_i \text{ 包含在 } m_r \text{ 中} \\ 0 & \text{其他} \end{cases}$$

注意 $on(1, m_4)=1$，$on(2, m_4)=1$ 和对于所有的 $i>2$，$on(i, m_4)=0$。最小项 m_1 表示了不包含任何索引项的文档中常见的共现模式（这可能发生，如当手动选择索引项集合时）。最小项 m_2 表示了仅含有索引项 k_1 的文档所特有的索引项共现模式。最小项 m_{2^t} 则表示了含有所有索引项的文档。

定义　对任意文档 d_j，存在一个最小项（或者合取分量）恰好包含出现在该文档中的所有索引项，并且不包含其他索引项。我们称这个最小项为 $c(d_j)$。

广义向量空间模型的中心思想是对每个最小项赋予一个单位向量 $\vec{m}_r$，并假设所有这样的单位向量组成的集合形成了一个正交向量基，如下所示。

定义　让我们定义如下的最小项向量 $\vec{m}_r$ 的集合：

$$1,2,\cdots,2^t$$
$$\vec{m}_1 = (1,0,\cdots,0)$$
$$\vec{m}_2 = (0,1,\cdots,0)$$
$$\vdots$$
$$\vec{m}_{2^t} = (0,0,\cdots,1)$$

对每个单位向量 $\vec{m}_r$ 分别赋予一个最小项 m_r。而且，对所有 $i \neq j$，$\vec{m}_i \cdot \vec{m}_j = 0$。

这样，根据定义，向量 $\vec{m}_r$ 的集合是两两正交的。那么，向量 $\vec{m}_r$ 的集合可以被看做是广义向量空间模型的正交基。 99

向量 $\vec{m}_r$ 之间的两两正交不表示索引项之间的独立性。相反，$\vec{m}_r$ 向量现在与索引项相关。例如，向量 $\vec{m}_4$ 对应的最小项是 $m_4=(1, 1, \cdots, 0)$，该最小项表示了文档集中含有索引项 k_1 和 k_2 而不包含其他索引项的文档。我们称最小项 m_4 引起了索引项 k_1 和 k_2 之间的关联性。如果我们更仔细地考虑这点，那么我们会注意到广义向量空间模型采用了如下的思想作为基础，即索引项在文档中的共现预示了这些词之间的关联性。这个模型的主要贡献是建立了一套正式的框架，可以很好地表示索引项之间的关联性。

为了确定索引项 k_i 的索引项向量 $\vec{k}_i$，我们简单地累加所有包含索引项 k_i 的最小项，并归一化。也就是，

$$\vec{k}_i = \frac{\sum_{\forall r} on(i,m_r) c_{i,r}\ \vec{m}_r}{\sqrt{\sum_{\forall r} on(i,m_r) c_{i,r}^2}} \tag{3-32}$$

$$c_{i,r} = \sum_{d_j \mid c(d_j)=m_r} w_{i,j}$$

其中，$w_{i,j}$ 是 TF-IDF 权重，可以用式（3-7）计算。

这些式子提供了一种用向量 $\vec{m}_r$ 表示索引项向量 $\vec{k}_i$ 的广义定义。索引项向量 $\vec{k}_i$ 由所有包含 k_i 的向量 $\vec{m}_r$ 组成。对每个 $\vec{m}_r$ 向量，定义相关系数 $c_{i,r}$ 为权重 $w_{i,j}$ 的和。每个 $w_{i,j}$ 权重与文

档 d_j 相关，该文档的索引项共现模式恰好和最小项 m_r 相符，即 $m_r=c(d_j)$。

仅当文档集中至少存在一篇文档符合最小项的索引项共现模式时，该最小项才值得关注，这种情况称为是激活的（active）。至多有 N 个最小项是激活的，其中 N 是文档集中文档的数量。因此，排序计算不像式（3-32）中那样需要数量为指数级别的最小项。

注意，内积 $\vec{k}_i \cdot \vec{k}_j$ 现在可以用来量化索引项 k_i 和 k_j 之间的相关程度。例如，

$$\vec{k}_i \cdot \vec{k}_j = \sum_{\forall r} on(i,m_r) \times c_{i,r} \times on(j,m_r) \times c_{j,r} \tag{3-33}$$

这是量化索引项相关性的好方法。

在经典向量模型中，文档 d_j 和用户查询 q 分别表示成 $\vec{d}_j = \sum_{\forall i} w_{i,j}\, \vec{k}_i$ 和 $\vec{q} = \sum_{\forall i} w_{i,q}\, \vec{k}_i$。在广义向量空间模型中，利用式（3-32）可以将这些表示形式直接转换为最小项向量 $\vec{m}_r$。于是，使用标准余弦相似度函数，可以用得到的 $\vec{d}_j$ 和 $\vec{q}$ 向量计算排序。

从广义向量空间模型计算得到的排序结合了标准的项-文档权重 $w_{i,j}$ 和相关系数 $c_{i,r}$。然而，采用项间相关性不一定得到更好的检索质量，我们并不清楚在哪种情况下广义模型优于经典向量模型。而且，广义模型计算排序的代价在大的文档集中是相当高的，因为在这种情
100 况下，激活的最小项（即那些需要用来计算 $\vec{k}_i$ 向量）的可能与文档集中文档的数量成正比。除了这些缺点外，从理论角度看，广义向量空间模型的确引入了很重要的新想法。

3.4.2 潜在语义索引模型

正如之前讨论的，通过索引项集合来总结文档和查询的内容会导致糟糕的检索质量，这归咎于两点。首先，许多不相关文档可能包含在答案集中。其次，无法检出未被查询中的关键词索引到的相关文档。造成这两点的主要原因是基于关键词集合的检索过程固有的模糊性。

文档的正文是叙述性的，涉及客观世界中的事物，我们称之为概念，及概念之间的关系。因此，把文档和给定的查询匹配可能基于概念匹配而不是索引项匹配。这使得文档即使没有被查询项索引到但仍可以被检出。例如，一篇文档可以被检出，因为它和另一篇与给定查询相关的文档共享了概念。潜在语义索引（Latent semantic indexing）是处理该问题的一种尝试。

潜在语义索引模型［614］的主要想法是把每篇文档向量和查询向量映射为由概念构成的维度空间。首先把索引项映射到概念维度空间上，然后利用这些映射关系来对文档和查询建模。其理由是在约化（降维）的概念空间上的检索可能要优于索引项空间上的检索。在论述之前，让我们定义基本术语。

定义 如前所述，设 t 是文档集内索引项的数量，N 是文档的总数量，且 $\boldsymbol{M}=[m_{i,j}]$ 是一个含有 t 行 N 列的项-文档矩阵。对这个矩阵的每个元素 $m_{i,j}$ 赋予一个项-文档对（k_i，d_j）的权重 $w_{i,j}$。这个权重 $w_{i,j}$ 可以用经典向量模型的 TF-IDF 权重来生成。

潜在语义索引建议用奇异值分解把 $\boldsymbol{M}$ 矩阵分解成三个部分，如下所示。

$$\boldsymbol{M} = \boldsymbol{K} \cdot \boldsymbol{S} \cdot \boldsymbol{D}^{\mathrm{T}}$$

矩阵 $\boldsymbol{K}$ 是项间相关性矩阵 $\boldsymbol{C}=\boldsymbol{M} \cdot \boldsymbol{M}^{\mathrm{T}}$ 的特征向量构成的矩阵（见 3.2.3 节对项间相关性矩阵的描述）。矩阵 $\boldsymbol{D}^{\mathrm{T}}$ 是由文档间矩阵 $\boldsymbol{M}^{\mathrm{T}} \cdot \boldsymbol{M}$ 特征向量所构成的矩阵的转置。矩阵 $\boldsymbol{S}$ 是由奇异值构成的 $r \times r$ 对角矩阵，其中 $r=\min(t, N)$，是 $\boldsymbol{M}$ 的秩。

现在考察仅使用 $\boldsymbol{S}$ 的前 s 个最大奇异值及其在 $\boldsymbol{K}$ 和 $\boldsymbol{D}^{\mathrm{T}}$ 中对应的列向量（即忽略 $\boldsymbol{S}$ 中其他的奇异值）。结果产生的 $\boldsymbol{M}_s$ 矩阵秩为 s，它从最小均方误差的角度看最接近于原始的 $\boldsymbol{M}$

矩阵。这个矩阵如下给出：

$$\boldsymbol{M}_s = \boldsymbol{K}_s \cdot \boldsymbol{S}_s \cdot \boldsymbol{D}_s^{\mathrm{T}}$$ 101

其中 s，$s<r$ 是约化概念空间的维度。对 s 值的选择可以通过平衡两个相反的效应来达到。首先，s 应该大到足以允许拟合所有真实数据的结构。其次，s 应该小到过滤掉所有不相关的（在传统的基于索引项的表式中呈现出来的）具体细节。

在维度为 s 的约化空间里，任意两篇文档的关系可以从如下给定的 $\boldsymbol{M}_s^{\mathrm{T}} \cdot \boldsymbol{M}_s$ 矩阵获得：

$$\begin{aligned}
\boldsymbol{M}_s^{\mathrm{T}} \cdot \boldsymbol{M}_s &= (\boldsymbol{K}_s \cdot \boldsymbol{S}_s \cdot \boldsymbol{D}_s^{\mathrm{T}})^{\mathrm{T}} \cdot \boldsymbol{K}_s \cdot \boldsymbol{S}_s \cdot \boldsymbol{D}_s^{\mathrm{T}} \\
&= \boldsymbol{D}_s \cdot \boldsymbol{S}_s \cdot \boldsymbol{K}_s^{\mathrm{T}} \cdot \boldsymbol{K}_s \cdot \boldsymbol{S}_s \cdot \boldsymbol{D}_s^{\mathrm{T}} \\
&= \boldsymbol{D}_s \cdot \boldsymbol{S}_s \cdot \boldsymbol{S}_s \cdot \boldsymbol{D}_s^{\mathrm{T}} \\
&= (\boldsymbol{D}_s \cdot \boldsymbol{S}_s) \cdot (\boldsymbol{D}_s \cdot \boldsymbol{S}_s)^{\mathrm{T}}
\end{aligned}$$

在上述矩阵中，(i, j) 元素量化了文档 d_i 和 d_j 间的关系。

给定用户查询，为了对文档排序，我们仅把查询建模为原始项-文档矩阵 $\boldsymbol{M}$ 中的一篇伪文档。假设将查询建模为编号为 0 的文档。那么，矩阵 $\boldsymbol{M}_s^{\mathrm{T}} \cdot \boldsymbol{M}_s$ 的第一行提供了所有文档对于查询的排序。

由于在潜在语义索引模型中使用的矩阵秩为 s，$s \ll t$，且 $s \ll N$，因此它们形成了对文档集中文档有效的索引方案。而且，提供了去除噪声（在基于索引项的表式中）和重复的方法。

潜在语义索引模型基于奇异值分解，引入了概念化信息检索问题的有趣方法。这体现了作为新理论框架的价值。但从实际的角度看，潜在语义索引还没有产生令人振奋的结果。

3.4.3　神经网络模型

在信息检索系统中，通过比较文档向量和查询向量，从而计算出排序。因此，为了计算排序，文档和查询中的索引项要被匹配和确定权重。神经网络是众所周知的、良好的模式匹配器，自然会被想到用做信息检索中的一种替代模型。

现在公认，我们的大脑是由数十亿的神经元组成。每个神经元可以被看做是一个基本的处理单元，当输入信号激励时，发射输出信号作为反应行为。由神经元发射的信号通过突触连接被馈入其他神经元，这些神经元可以再发射新的输出信号。这个过程本身可以在多层神经元上重复，并通常称为*激活扩散过程*。因此，输入的信息被处理了（即被分析和解释了），这可使大脑要求做出实际的反应（比如肌肉动作）作为回应。

人类大脑的神经元网格纵横交错，神经网络是对其极大简化后获得的图表示形式。图中的结点是处理单元，而边则起到了突触连接的作用。为了模仿人类大脑内突触的强度随时间变化的现象，神经网络中的每条边都被赋予了一个权重。 102 在每个瞬间，结点的状态由它的激发水平（这是一个关于其初始状态和收到的输入信号的函数）来定义。根据它的激发水平，结点 A 可能发送一个信号给结点 B。在结点 B 的信号强度依赖于结点 A 和 B 边上的权重。

信息检索的神经网络的定义如图 3-13 所示。这里描述的模型是基于［1697］中的工作。

我们首先观察到图 3-13 中的神经网络由三

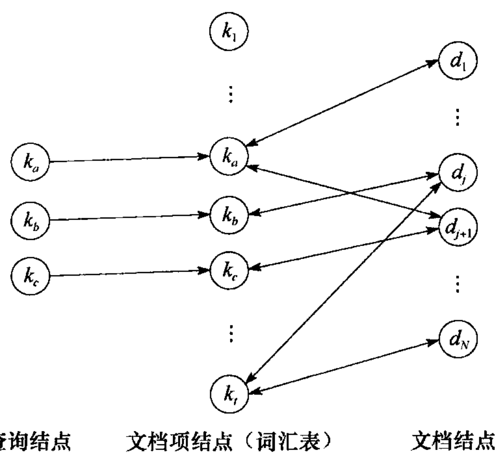


图 3-13　信息检索的神经网络模型

层构成：第一层是查询项，第二层是文档项，第三层是文档本身。注意这个神经网络的拓扑结构和图 3-15 中推理网的拓扑结构之间的相似性。然而，这里是由查询项结点发送信号给文档项结点来初始化推理过程的。接着，文档项结点本身可能产生信号到文档结点。第一阶段就完成了，信号从查询项结点传输到了文档结点（即在图 3-13 中从左往右）。

然而，神经网络没有在第一阶段信息传播后停止运作。可能轮到文档结点产生新的信号直接发送到文档项结点（这就是文档和文档项结点间是双向边的原因）。在收到这个激励后，文档项结点可能又发射新的信号直接送到文档结点，并重复这个过程。在每次迭代过程后，信号变得越来越弱，激活扩散过程最终停止。这个过程可能激活了文档 d_l，即使它不包含任何查询项。因此，整个过程可以解释为激活了内建的同义词典。

103 查询项结点被赋予了一个最大值为 1 的初始（并固定的）激发水平。然后，查询项结点发送信号到文档项结点，该信号被归一化查询项权重 $\overline{w}_{i,q}$ 削弱。对于基于向量的排序，该权重可以从为向量模型定义的权重 $w_{i,q}$ 获得，例如：

$$\overline{w}_{i,q} = \frac{w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,q}^2}}$$

其中归一化是通过查询向量的范数来进行的。

当信号抵达文档项结点时，它们可能直接传送新的信号给文档结点。这些信号被归一化文档项权重 $\overline{w}_{i,j}$ 削弱。该权重可以由式（3-7）定义的向量模型的权重 $w_{i,j}$ 产生，例如：

$$\overline{w}_{i,j} = \frac{w_{i,j}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2}}$$

其中归一化是通过文档向量的范数进行的。

到达文档结点的信号被累加起来。因此，在首轮信号传播后，文档 d_j 对应的文档结点的激发水平按下式给出

$$\sum_{i=1}^{t} \overline{w}_{i,q}\overline{w}_{i,j} = \frac{\sum_{i=1}^{t} w_{i,q} w_{i,j}}{\sqrt{\sum_{i=1}^{t} w_{i,q}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,j}^2}} \qquad (3\text{-}34)$$

这正是经典向量模型提供的排序。

为了提高检索质量，神经网络在首轮传播后接着进行激活扩散过程。这改变了初始向量的排序，类似于用户相关反馈循环（见第 5 章）。为了使这个过程更有效，应该定义一个最小激活阈值，使得低于该阈值的文档结点不再发送信号。详细的情况可以在［1697］中找到。

没有确实的证据表明神经网络在一般文档集上有更好的检索质量。实际上，该模型还没有在大的文档集上广泛测试过。然而，神经网络给出了一种建模的新范式。而且，它使得可以自然地检索出最初和查询项没有关联的文档——这是一项很吸引人的功能。

3.5 其他概率模型

本节讨论四种其他的概率模型：1）BM25 模型，这是经典概率模型的延伸；2）语言模型，这是使用文档集中索引项的概率分布作为排序基础这一想法的现代变体；3）随机差异模型；4）信念网模型，这是贝叶斯网在信息检索中的直接应用。

3.5.1 BM25 模型

BM25 模型是在标准概率公式（3-17）的变体上经过一系列实验诞生的。这些实验是出于这样的观察，即在经典向量模型中，好的索引项权重计算基于三个原则：1）反比文档频率；2）项频；3）文档长度归一化。式（3-17）的标准概率公式涵盖了这些原则的第一条，但没包含其他两条。所以，用项频和文档长度归一化来扩展它看起来是一种提高结果的自然方法。这个推理导致了在 Okapi 系统 [1366，1367，1368] 上的一系列实验，并产生了 BM25 排序公式。 104

1. BM1、BM11 和 BM15 排序公式

最初，Okapi 系统使用式（3-17）作为排序公式 [1366]。这称为 BM1 公式，缩写 BM 表示 Best Match（最佳匹配）。

改进排序最初的想法是在式（3-17）中引入一个项频因子。该因子在一些转变后，演变为：

$$\mathcal{F}_{i,j} = S_1 \times \frac{f_{i,j}}{K_1 + f_{i,j}} \tag{3-35}$$

其中 $f_{i,j}$ 是索引项 k_i 在文档 d_j 的频率，K_1 是通过在具体的文档集上实验获得的常数，S_1 是和 K_1 相关的尺度常数，通常设为 $S_1=(K_1+1)$。注意，如果设 $K_1=0$，那么这参数变成 1，对排序没有影响。

下一步是把文档归一化引入公式中。这可以通过将上述的式子改变为：

$$\mathcal{F}'_{i,j} = S_1 \times \frac{f_{i,j}}{\frac{K_1 \times len(d_j)}{avg_doclen} + f_{i,j}} \tag{3-36}$$

其中 $len(d_j)$ 是文档 d_j 的长度（例如文档内索引项的数量），avg_doclen 是文档集的平均文档长度。

除了往 TF 因子中引入上述所示的文档长度归一化外，接下来的方案是增加一个依赖于文档和查询长度的修正因子 $\mathcal{G}_{j,q}$：

$$\mathcal{G}_{j,q} = K_2 \times len(q) \times \frac{avg_doclen - len(d_j)}{avg_doclen + len(d_j)} \tag{3-37}$$

其中，$len(q)$ 是查询长度（查询内索引项的个数），且 K_2 是常数。这个因子不依赖于和文档 d_j 匹配的具体查询项，仅依赖于文档和查询的长度。这是一个全局因子，接下来会加以简单的介绍。

类似的推理可以被应用到查询的项频上，从而得到一个额外的因子：

$$\mathcal{F}_{i,q} = S_3 \times \frac{f_{i,q}}{K_3 + f_{i,q}} \tag{3-38}$$

其中，$f_{i,q}$ 是索引项 k_i 在查询 q 中的频率，K_3 是常数，且 S_3 是和 K_3 相关的尺度常数，通常设为 $S_3=(K_3+1)$。 105

把这三个因子引入到式（3-17）中可获得不同的 BM（即 Best Matching）公式，如下所示。

$$sim_{BM1}(d_j, q) \sim \sum_{k_i[q,d_j]} \log\left(\frac{N - n_i + 0.5}{n_i + 0.5}\right)$$

$$sim_{BM15}(d_j, q) \sim \mathcal{G}_{j,q} + \sum_{k_i[q,d_j]} \mathcal{F}_{i,j} \times \mathcal{F}_{i,q} \times \log\left(\frac{N - n_i + 0.5}{n_i + 0.5}\right)$$

$$sim_{BM11}(d_j,q) \sim \mathcal{G}_{j,q} + \sum_{k_i[q,d_j]} \mathcal{F}'_{i,j} \times \mathcal{F}_{i,q} \times \log\left(\frac{N-n_i+0.5}{n_i+0.5}\right)$$

其中 $k_i[q, d_j]$ 是 $k_i \in q \wedge k_i \in d_j$ 的简短记号。在［1373］中的经验证据表明，最好取 $K_2=0$，这可以从公式中消去修正因子$\mathcal{G}_{j,q}$。而且，尺度常数的恰当估计值是 $S_1=(K_1+1)$ 和 $S_3=(K_3+1)$，经验证据表明，使 K_3 非常大更好［1373］。在这种情况下，因子$\mathcal{F}_{i,q}$简单地简化为 $f_{i,q}$。对于短查询，假设对于所有索引项 $f_{i,q}$是1。这些想法导致了如下更简单的公式。

$$sim_{BM1}(d_j,q) \sim \sum_{k_i[q,d_j]} \log\left(\frac{N-n_i+0.5}{n_i+0.5}\right) \tag{3-39}$$

$$sim_{BM15}(d_j,q) \sim \sum_{k_i[q,d_j]} \frac{(K_1+1)f_{i,j}}{(K_1+f_{i,j})} \times \log\left(\frac{N-n_i+0.5}{n_i+0.5}\right)$$

$$sim_{BM11}(d_j,q) \sim \sum_{k_i[q,d_j]} \frac{(K_1+1)f_{i,j}}{\frac{K_1\ len(d_j)}{avg_doclen}+f_{i,j}} \times \log\left(\frac{N-n_i+0.5}{n_i+0.5}\right)$$

这些是在BM1、BM15和BM11排序公式中实际使用的公式。使用TREC数据获得的经验结果表明BM11一致地优于BM15。考虑到文档长度归一化对排序的重要性，这并不奇怪。

2. BM25 排序公式

BM25是结合了BM11和BM25排序公式产生的。其动机是按如下方式结合项频。

$$\mathcal{B}_{i,j} = \frac{(K_1+1)f_{i,j}}{K_1\left[(1-b)+b\frac{len(d_j)}{avg_doclen}\right]+f_{i,j}} \tag{3-40}$$

其中 b 是新引入的常数，其值位于区间［0，1］中。若 $b=0$，上述该式子简化为BM15项频因子；若 $b=1$，它简化为BM11项频因子；当 b 的值介于 0～1 之间时，该式子提供了BM11和BM15的一种结合。

BM25模型的排序公式可以写为：

106

$$sim_{BM25}(d_j,q) \sim \sum_{k_i[q,d_j]} \mathcal{B}_{i,j} \times \log\left(\frac{N-n_i+0.5}{n_i+0.5}\right) \tag{3-41}$$

其中表达式$\mathcal{B}_{i,j}$中的 K_1 和 b 是根据经验确定的常数。举个例子来说，$K_1=1$ 是一个在实际文档集上运作良好的合理假设［1373］，而 b 应该接近1来强调BM11公式中文档长度归一化的效果。比方说，$b=0.75$ 就是一个合理的假设［1368］。最重要的是，这些参数的值能通过适当的实验针对具体文档集进行精细的调整。

式（3-41）如今是BM25模型的排序公式［1370］。与原始的经典概率模型中的概念不同，BM25公式能在完全自动的模式（用户不提供相关信息）下计算。而且，更多的数据表明一旦进行了精细的调节，该模型在一般文档集上可以产生比经典向量模型更好的结果。因此，它取代了向量模型，被用做评价新的排序方法的基准。

3.5.2 语言模型

语言模型（language model）用于许多自然语言处理应用，例如词性标注、语音识别、机器翻译和信息检索。举例来说，从事语音识别的研究人员定义概率分布来对口语中的规律性建模，并用它们来预测序列中下一个符号是某个特定词的可能性。这些概率分布称为语言模型［1384］。

语言模型定义了文档的概率分布并用它们来预测观察到查询项的可能性，从而推动了信

息检索模型的研究。其想法是为文档集中的每篇文档定义语言模型，并用它来获得给定查询的似然度。也就是说，不是用查询去预测观察到的文档的似然度，而是用文档的正文去预测观察到查询的概率。通过对这些概率值排序可以产生文档的排序。

把语言模型应用到信息检索中首先是由 Kalt[864] 提出的。Ponte 和 Croft[1290] 随后开展了相应的工作，他们进行了广泛的实验，证明了语言模型优于经典向量模型等基准模型。因此，Ponte 和 Croft 的工作被认为是在信息检索中运用语言模型的里程碑，在后续的章节中要谈到。然而，后来许多研究人员证明 Kalt 原来的方法更好。因此，我们首先从这种方法的统计基础开始。

1. 统计基础

在深入研究信息检索的语言模型之前，让我们简短地回顾该方法的统计基础。我们的讨论是基于 Liu 和 Croft 的极好的综述 [1042]。

设 S 是由出现在文档集中同一篇文档中的 r 个连续的索引项构成的序列。也就是，

$$S = k_1, k_2, \cdots, k_r$$ 107

n 元语言模型使用马尔科夫过程对索引项序列 S 的出现赋予一个概率值，如下所示。

$$P_n(S) = \prod_{i=1}^{r} P(k_i | k_{i-1}, k_{i-2}, \cdots, k_{i-(n-1)})$$

其中 n 是马尔科夫过程的阶。该分布基本上是多项分布，其中某个索引项出现的概率依赖于文中出现在它之前的 $n-1$ 个索引项的情况。若 $n=2$，则我们有了一个二元语言模型，因为参数估计基于一对词语之间的共现信息。若 $n=1$，则我们有了一个一元语言模型，因为参数估计基于词语单独出现的概率。

对于像语音识别和机器翻译这样的应用，词语的顺序是非常重要的。因此，通常采用三元语言模型等高阶模型。这些模型计算代价更加昂贵。举例来说，估计索引项 k_i 在一元语言模型中的概率只需要计算 $P(k_i)$。在三元语言模型中，相同的估计需要计算文档集中所有以 k_i 结尾的三元组，即所有形如 $P(k_i | k_{i-1}, k_{i-2})$ 的概率。幸好，在信息检索的情况下，词序的影响是不明显的。因此，一元语言模型已经被广泛使用（即索引项的独立性假设），正如我们在本章大部分中做的那样。

2. 基于多项过程的语言模型

语言模型中的排序是通过估计 $P(q|M_j)$ 获得的。Ponte 和 Croft[1290] 采用了伯努利过程来产生查询，而 Hiemstra[760]、Miller、Leek 和 Schwartz[1132]，以及 Song 和 Croft[1502] 等其他研究人员，则采用了最初由 Kalt[864] 提出的多项过程。现在它是信息检索中由语言模型生成查询的标准过程。根据这个过程，如果假设索引项之间是独立的（一元模型），那么我们就能获得（我们的讨论极大地受到 [1772] 的影响）：

$$P(q|M_j) = \prod_{k_i \in q} P(k_i | M_j) \tag{3-42}$$

在两边取 log 函数：

$$\begin{aligned} \log P(q|M_j) &= \sum_{k_i \in q} \log P(k_i | M_j) \\ &= \sum_{k_i \in q \wedge d_j} \log P_{\in}(k_i | M_j) + \sum_{k_i \in q \wedge \neg d_j} \log P_{\notin}(k_i | M_j) \\ &= \sum_{k_i \in q \wedge d_j} \log\left(\frac{P_{\in}(k_i | M_j)}{P_{\notin}(k_i | M_j)}\right) + \sum_{k_i \in q} \log P_{\notin}(k_i | M_j) \end{aligned} \tag{3-43}$$

其中 $P_{\in}$ 和 $P_{\notin}$ 是两个不同的概率分布。前者是查询项在文档中的分布，而后者是查询项不

[108] 在文档中的分布。对于第二个分布，考虑到该查询项不在文档 d_j 中，其统计量是从所有的文档集上获得。因此有：

$$P_{\notin}(k_i|M_j) = \alpha_j P(k_i|C) \tag{3-44}$$

其中 α_j 是对应于文档 d_j 的参数，$P(k_i|C)$ 是文档集 C 的语言模型，例如式（3-49）所定义的。

把式（3-44）代入式（3-43）中，得到：

$$\log P(q|M_j) = \sum_{k_i \in q \wedge d_j} \log\left(\frac{P_{\in}(k_i|M_j)}{\alpha_j P(k_i|C)}\right) + n_q \log \alpha_j + \sum_{k_i \in q} \log P(k_i|C)$$

$$\sim \sum_{k_i \in q \wedge d_j} \log\left(\frac{P_{\in}(k_i|M_j)}{\alpha_j P(k_i|C)}\right) + n_q \log \alpha_j \tag{3-45}$$

其中 n_q 表示查询长度，最后的累加和被舍弃了，因为它对所有文档 d_j 是常数。排序函数现在由两个独立的部分构成。第一个部分给每个出现在文档中的查询项赋予权重，根据表达式

$$\log\left(\frac{P_{\in}(k_i|M_j)}{\alpha_j P(k_i|C)}\right)$$

该项权重直接与文档中的项频成正比，与文档集中的项频成反比，这和向量模型中 IDF 权重的作用类似。其次，参数 α_j 可以用于文档长度归一化。第二部分由 $n_q \log \alpha_j$ 给定，给不出现在文档中的查询项赋予一小部分概率块。

一项重要的观察是，结合多项过程和平滑方法来生成查询，会自然地获得一个包含项频、IDF 和文档归一化的排序函数，这与之前的概率模型不同。也就是说，平滑在现代语言模型中扮演着重要的作用。

3. 平滑

在我们之前的讨论中，我们用 $P(k_i|C)$ 估计 $P_{\notin}(k_i|M_j)$，即用整个文档集的统计量来避免给不出现在文档中的索引项赋予零概率。估计 $P_{\notin}(k_i|M_j)$ 的过程对精细地调节语言模型的排序公式是重要的，它称为平滑。

一个流行的平滑技术是把文档中某些查询项的概率块移到不出现在文档中的索引项上 [366]。这可以通过修改式（3-42）中的 $P(k_i|M_j)$ 来实现，如下所示。

$$P(k_i|M_j) = \begin{cases} P^s_{\in}(k_i|M_j) & k_i \in d_j \\ \alpha_j P(k_i|C) & \text{其他} \end{cases}$$

其中 $P^s_{\in}(k_i|M_j)$ 是文档 d_j 中索引项的平滑分布。由于 $\sum_i P(k_i|M_j) = 1$，因此有

[109]

$$\sum_{k_i \in d_j} P^s_{\in}(k_i|M_j) + \sum_{k_i \notin d_j} \alpha_j P(k_i|C) = 1$$

也就是，

$$\alpha_j = \frac{1 - \sum_{k_i \in d_j} P^s_{\in}(k_i|M_j)}{1 - \sum_{k_i \in d_j} P(k_i|C)} \tag{3-46}$$

在上述假设下[⊖]，平滑参数 α_j 也是 $P^s_{\in}(k_i|M_j)$ 的函数。于是，通过对 $P^s_{\in}(k_i|M_j)$ 的不同定义可以获得不同的平滑方法，我们接下来要加以讨论。更多平滑方法的讨论可见 [1772]。

（1）使用 Jelinek-Mercer 的平滑方法

这种方法的思想是在文档频率和文档集频率分布之间进行线性插值，如下所示。

⊖ 注意 $\sum_{k_i \in d_j} P^s_{\in}(k_i|M_j) < 1$，因为有些概率块 $P_{\in}(k_i|M_j)$ 已经被转移到不在文档 d_j 中的索引项。

$$P_{\in}^{s}(k_i|M_j)=(1-\lambda)\frac{f_{i,j}}{\sum_i f_{i,j}}+\lambda\frac{F_i}{\sum_i F_i}$$

其中 $0\leqslant\lambda\leqslant 1$ 是必须被经验定义的参数。λ 越接近于 0，索引项的文档频率的影响力就越大。当 λ 向 1 变动时，索引项的文档集总频率的影响力就越大，即平滑的效应越大。通过把 $P_{\in}^{s}(k_i|M_j)$ 代入式（3-46）中，可以证明 $\alpha_j=\lambda$。

（2）使用 Dirichlet 先验的贝叶斯平滑

这种方法也称为 Dirichlet 平滑，其中的语言模型是多项分布，而它的共轭先验概率密度符合 Dirichlet 分布。于是有：

$$P_{\in}^{s}(k_i|M_j)=\frac{f_{i,j}+\lambda\frac{F_i}{\sum_i F_i}}{\sum_i f_{i,j}+\lambda}$$

如前所述，λ 越接近于 0，索引项的文档频率的影响力就越大。λ 向 1 变动时，索引项的文档集频率的影响力就增大了。但是，和之前的情况相反，它只是和文档频率部分混合。如前所述，λ 的值越大，平滑的效应就越大。通过把 $P_{\in}^{s}(k_i|M_j)$ 代入式（3-46）中，可以证明 $\alpha_j=\lambda\Big/\Big(\sum_i f_{i,j}+\lambda\Big)$。

（3）平滑计算的效率

在上述两种平滑方法中，可以高效地执行计算。所有的频率计数可以直接通过索引获得，也可以为每篇文档预先计算 α_j 的值。因此，其复杂性类似于使用 TF-IDF 权重的向量空间排序。 110

（4）把平滑应用到信息检索排序中

多项语言模型中的信息检索排序是用式（3-45）计算的，如下所示。

- 使用某种平滑方法计算 $P_{\in}^{s}(k_i|M_j)$（上面讨论的两种方法中的一种，或文献中的其他方法）。
- 使用式（3-49）计算 $P(k_i|C)$。
- 用刚计算出的 $P_{\in}^{s}(k_i|M_j)$ 和 $P(k_i|C)$ 的值，使用式（3-46）计算 α_j。
- 用 $P_{\in}^{s}(k_i|M_j)$ 代替 $P(k_i|M_j)$，计算式（3-45）的排序。

我们注意到平滑在排序公式中起着关键的作用，直接影响结果的质量，这在［1772］中被广泛讨论。

4. 基于伯努利过程的语言模型

在 Ponte 和 Croft［1290］的工作中，提出了如下的伯努利过程。

定义 给定文档 d_j，设 M_j 表示该文档的参考语言模型。这个语言模型允许从模型估计生成用户查询 q 的概率，即估计条件概率 $P(q|M_j)$。

如果我们假设索引项之间的独立性，那么我们能用多元伯努利过程计算 $P(q|M_j)$，如下所示。

$$P(q|M_j)=\prod_{k_i\in q}P(k_i|M_j)\times\prod_{k_i\notin q}P(k_i|M_j) \tag{3-47}$$

其中 $P(k_i|M_j)$ 是索引项的概率。这类似于经典概率模型中计算排序的表达式。

索引项概率的简单估计是：

$$P(k_i|M_j)=\frac{f_{i,j}}{\sum_i f_{i,j}}$$

它简单地计算了从文档 d_j 的索引项集合中随机抽取索引项 k_i 的概率。然而，这个简化的公式有问题。如果查询项不出现在文档中（即不允许部分匹配），那么概率会变成零。为了克服这个限制，我们推测，某个索引项不出现在文档中不能表明该索引项和文档之间没有联系。相反地，我们假设未出现的索引项和文档有一定关联，其概率 $P(k_i|C)$ 是在整个文档集 C 中观察该项的概率。

概率 $P(k_i|C)$ 可以用不同的方法计算。例如 Hiemstra[760] 提出：

111

$$P(k_i|C) = \frac{n_i}{\sum_i n_i} \tag{3-48}$$

其中 n_i 是包含索引项 k_i 的文档的个数。像 IDF 一样，它对 $P(k_i|C)$ 进行了估计。Miller、Leek 和 Schwartz [1132] 则提出：

$$P(k_i|C) = \frac{F_i}{\sum_i F_i} \tag{3-49}$$

其中，

$$F_i = \sum_j f_{i,j}$$

该 $P(k_i|C)$ 公式估计了在文档集中观察到索引项 k_i 的最大似然度，这里采纳该公式。于是按如下公式重新定义 $P(k_i|M_j)$：

$$P(k_i|M_j) = \begin{cases} \dfrac{f_{i,j}}{\sum_i f_{i,j}} & f_{i,j} > 0 \\ \dfrac{F_i}{\sum_i F_i} & f_{i,j} = 0 \end{cases}$$

这个表达式仍然有一个基本的缺点——索引项的概率估计基于由单一的文档 d_j 构成的小样本。这显然不是我们所希望的，因为这会导致模型的不稳定。为了让模型更灵活，我们需要基于更大的文档样本进行估计[⊖]。这可以通过计算平均值达到，如下所示。

$$P(k_i) = \frac{\sum_{j|k_i \in d_j} P(k_i|M_j)}{n_i}$$

也就是，$P(k_i)$ 是基于所有包含索引项 k_i 的文档构成的语言模型来估计的。因为从更大的文档库中获得，这是一个更加稳定的估计量。然而，它对所有包含索引项 k_i 的文档都是一样的。也就是说，使用平均概率 $P(k_i)$ 来预测通过特定文档 d_j 的语言模型产生索引项 k_i 的概率是有风险的。

为了修正这个问题，定义索引项 k_i 在文档 d_j 上的平均频率 $\overline{f}_{i,j}$ 为

$$\overline{f}_{i,j} = P(k_i) \times \sum_i f_{i,j}$$

也就是说，如果索引项在文档中的出现是由平均概率 $P(k_i)$ 控制的，则 $\overline{f}_{i,j}$ 估计了索引项 k_i 在文档 d_j 的出现频率。使用 $\overline{f}_{i,j}$，而不是直接使用 $f_{i,j}$，作为 k_i 在 d_j 中频率的估计，其风险 $R_{i,j}$ 可以用如下的几何分布来量化。

112

$$R_{i,j} = \left(\frac{1}{1+\overline{f}_{i,j}}\right) \times \left(\frac{\overline{f}_{i,j}}{1+\overline{f}_{i,j}}\right)^{f_{i,j}}$$

⊖ 概率估计即使在大样本中也可能变困难，因为短语可能为任意长度，并因此在样本中无法观察到。

对于文档集中经常出现的索引项，$\bar{f}_{i,j} \gg 0$ 且 $R_{i,j} \sim 0$。对于在文档和文档集中都稀少的项，$f_{i,j} \sim 1$，$\bar{f}_{i,j} \sim 1$ 且 $R_{i,j} \sim 0.25$。

平均概率 $P(k_i)$ 的风险被用作为一种混合参数，该参数允许在由更大的文档样本构成的语言模型 M_j 中，更好地估计索引项 k_i 出现的概率。让我们把这个概率称为 $P_R(k_i|M_j)$。然后我们用风险因子 $R_{i,j}$ 计算 $P_R(k_i|M_j)$。

$$P_R(k_i|M_j) = \begin{cases} P(k_i|M_j)^{(1-R_{i,j})} \times P(k_i)^{R_{i,j}} & f_{i,j} > 0 \\ \dfrac{F_i}{\sum_i F_i} & \text{其他} \end{cases}$$

在这个公式中，如果 $R_{i,j} \sim 0$，那么 $P_R(k_i|M_j)$ 主要是 $P(k_i|M_j)$ 的函数；否则，它混合了 $P(k_i)$ 和 $P(k_i|M_j)$。代入式（3-47）中，得到：

$$P(q|M_j) = \prod_{k_i \in q} P_R(k_i|M_j) \times \prod_{k_i \notin q} [1 - P_R(k_i|M_j)] \tag{3-50}$$

该式计算了从文档语言模型生成查询的概率。

3.5.3 随机差异模型

Amati 和 Rijsbergen [39] 提出了一个独特的概率模型，它具有语言模型的某些特点。其思想是通过度量由随机过程产生的索引项分布和真实索引项分布之间的差异（divergence）来计算索引项的权重。因此，其名字是*随机差异模型*（divergence from randomness）。该模型基于两个基本假设 [7]，如下所示。

- 不是所有的词语在描述文档内容上都是同等重要的。含有少量信息的词语在整个文档集 C 内是随机分布的。给定索引项 k_i，其在整个文档集上的概率分布是 $P(k_i|C)$，其对应的信息量由 $-\log P(k_i|C)$ 给出。通过对这个分布的不同定义，能获得索引项在文档集中不同的*随机度概念*。
- 索引项的补充分布可以通过仅考察含有 k_i 的文档子集获得。这个子集称为*精华集*（elite set）。给定文档 d_j，对应的概率分布记做 $P(k_i|d_j)$。在文档 d_j 中观察到 k_i 的概率越小，这个索引项就被认为是越稀少、越重要。因此，这个索引项在精华集中带有的信息量被定义为 $1-P(k_i|d_j)$。

给定这些假设，索引项 k_i 在文档 d_j 中的权重 $w_{i,j}$ 定义为

$$w_{i,j} = (-\log P(k_i|C)) \times (1 - P(k_i|d_j)) \tag{3-51}$$ 113

正如在平滑中一样，随机差异模型同时考察了索引项在文档集中的分布和索引项在其出现的文档子集中的分布。文档 d_j 对于查询 q 的排序 $R(d_j, q)$ 可以计算为

$$R(d_j, q) = \sum_{k_i \in q} f_{i,q} \times w_{i,j} \tag{3-52}$$

其中，$f_{i,q}$ 是索引项 k_i 在查询中的频率。

1. 整个文档集上的随机分布

为了计算索引项在文档集上的分布，可以考虑不同的概率模型，例如用伯努利实验来对索引项在文档集中出现的情况建模。假设含有 1000 篇文档的文档集和在该文档集中出现 10 次以上的索引项 k_i，那么在文档中观察到索引项 k_i 出现 4 次的概率是

$$P(k_i|C) = \binom{10}{4}\left(\frac{1}{1000}\right)^4\left(1 - \frac{1}{1000}\right)^6$$

这是标准的二项分布。

通常，设 $p=1/N$ 是在文档中观察到索引项的概率，其中 N 是文档集中的总文档数。在文档 d_j 中观察到索引项 k_i 出现 $f_{i,j}$次的概率由如下所示的二项分布描述。

$$P(k_i|C) = \binom{F_i}{f_{i,j}} p^{f_{i,j}} \times (1-p)^{F_i - f_{i,j}} \tag{3-53}$$

定义

$$\lambda_i = p \times F_i$$

并假设当 $N \to \infty$时 $p \to 0$，但 $\lambda_i = p \times F_i$ 保持不变。在这样的条件下，能把二项分布近似为泊松过程，这样得到：

$$P(k_i|C) = \frac{e^{-\lambda_i} \lambda_i^{f_{i,j}}}{f_{i,j}!}$$

那么，在文档集中索引项 k_i 的信息量可以按如下计算：

114

$$\begin{aligned} -\log P(k_i|C) &= -\log\left(\frac{e^{-\lambda_i}\lambda_i^{f_{i,j}}}{f_{i,j}!}\right) \\ &\approx -f_{i,j}\log\lambda_i + \lambda_i \log e + \log(f_{i,j}!) \\ &\approx f_{i,j}\log\left(\frac{f_{i,j}}{\lambda_i}\right) + \left(\lambda_i + \frac{1}{12f_{i,j}+1} - f_{i,j}\right)\log e + \frac{1}{2}\log(2\pi f_{i,j}) \end{aligned} \tag{3-54}$$

其中的 log 函数以 2 为底，阶乘项 $f_{i,j}!$ 是由 Stirling 公式近似的：

$$f_{i,j}! \approx \sqrt{2\pi} f_{i,j}^{(f_{i,j}+0.5)} e^{-f_{i,j}} e^{(12f_{i,j}+1)^{-1}}$$

一种不同的方法是采用 Bose-Einstein 分布，并用几何分布来近似它，于是有：

$$P(k_i|C) \approx p \times p^{f_{i,j}}$$

其中 $p=1/(1+\lambda_i)$。索引项 k_i 在文档集中对应的信息量可以计算为：

$$-\log P(k_i|C) \approx -\log\left(\frac{1}{1+\lambda_i}\right) - f_{i,j} \times \log\left(\frac{\lambda_i}{1+\lambda_i}\right) \tag{3-55}$$

这提供了计算索引项在整个文档集上分布的第二种方法。

2. 精华集上的分布

为了计算精华文档上索引项的分布所对应的信息量，可以应用拉普拉斯（Laplace）连续定律，这样可得：

$$1 - P(k_i|d_j) = \frac{1}{f_{i,j}+1} \tag{3-56}$$

另一种可能是采用两个伯努利过程的比率，于是有：

$$1 - P(k_i|d_j) = \frac{F_i + 1}{n_i \times (f_{i,j}+1)} \tag{3-57}$$

其中 n_i 是出现索引项的文档数量，如前所示。

3. 归一化

这些公式没有考虑文档 d_j 的长度。这可以通过对项频 $f_{i,j}$进行归一化来实现。可以使用不同的归一化方法，例如：

$$f'_{i,j} = f_{i,j} \times \frac{avg_doclen}{len(d_j)}$$

或者

$$f'_{i,j} = f_{i,j} \times \log\left(1 + \frac{avg_doclen}{len(d_j)}\right)$$

其中 avg_doclen 是文档集中的平均文档长度，$len(d_j)$ 是文档 d_j 的长度。为了使用归一化项频计算项权重 $w_{i,j}$，只需要把频率因子 $f_{i,j}$替换为它的归一化形式 $f'_{i,j}$。而用于计算 $P(k_i|C)$

的归一化技术可以和计算 $P(k_i|d_j)$ 的归一化技术不同，这里我们考虑使用相同的归一化技术。 115

通过结合计算 $P(k_i|C)$ 和 $P(k_i|d_j)$ 的不同方法和不同的归一化技术，可以产生不同的随机差异排序公式。要完整地讨论这些排序公式，除了这里出现的公式外，还要考虑其他分布，读者可参考 [39]。

3.5.4 贝叶斯网模型

一种改良信息检索中概率模型的方法是使用贝叶斯信念网（Bayesian belief networks）。信念网是值得关注的，因为它提供了一套清晰的形式，结合了给定文档的不同证据源（如过去的查询、过去的反馈循环和不同的查询式）。更重要的是，结合这些不同证据源可以改进结果，如 Turtle 和 Croft[1610] 所展示的。

这里，我们讨论由 Turtle 和 Croft[1610] 提出的*推理网模型*。它提供了 Inquery 系统 [280] 中搜索引擎的理论基础，以及由 Ribeiro-Neto 和 Muntz 提出的称为*信念网模型*的变体 [1352]。

我们这里的讨论使用与 Turtle 和 Croft 原文不同的风格。尤其是，我们更关注模型改良中概率方面的讨论。为了推动主要的设计决策，我们特意回到贝叶斯形式上。我们认为这种方法加深了对精妙之处的理解。在此之前，我们简单地介绍贝叶斯网。

1. 贝叶斯网

贝叶斯网 [1251] 是有向无环图（Directed Acyclic Graph，DAG），其中的结点代表随机变量，有向边表示这些变量之间的因果关系，并且这些因果关系的强度是由条件概率表示的。某个结点的所有父结点（这个结点本身是子结点），被认为是它的*直接原因*。这种因果关系是由有向无环图中从父结点直接指向子结点的连接表示的。网络的*根*没有父结点。

设 x_i 是贝叶斯网 G 的一个结点，Γ_{x_i} 是 x_i 的父结点的集合。Γ_{x_i} 对于 x_i 的影响是由任何满足下述条件的一组函数 $F_i(x_i, \Gamma_{x_i})$ 定义的：

$$\sum_{\forall x_i} F_i(x_i,\Gamma_{x_i}) = 1 \quad \text{同时} \quad 0 \leqslant F_i(x_i,\Gamma_{x_i}) \leqslant 1$$

其中 x_i 也表示了结点 x_i 对应的随机变量的状态。这个定义是完备且一致的，因为积 $\prod_{\forall i} F_i(x_i,\Gamma_{x_i})$ 组成了一个 G 内结点的联合概率分布。

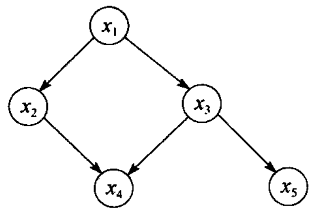


图 3-14 贝叶斯网的例子

图 3-14 说明了联合概率分布 $P(x_1, x_2, x_3, x_4, x_5)$ 的贝叶斯网。在这种情况下，网络中显示的依赖关系允许以局部条件概率表达联合概率分布（贝叶斯网的主要优点），这是一种自然的表达形式，如下所示。 116

$$P(x_1,x_2,x_3,x_4,x_5) = P(x_1)P(x_2|x_1)P(x_3|x_1)P(x_4|x_2,x_3)P(x_5|x_3)$$

概率 $P(x_1)$ 称为贝叶斯网的先验概率，能用来对应用中关于语义的先验知识进行建模。

2. 推理网模型

在概率论中，两派最传统的阵营是基于*频率论观点*（frequetist view）和基于*认识论观点*（epistemological view）的。频率论观点把概率看做是和偶然性规律相关的统计概念。认识论观点把概率解释为某种程度的置信度，其设定可能不来自统计性实验。第二种观点是重要的，因为我们在日常生活中经常在没有清楚定义产生概率的统计实验的情况下使用了概率这个概念。

推理网模型［1610，1609］采用了信息检索问题中的认识论观点。对于索引项、文档和用户查询赋予随机变量。文档 d_j 的随机变量表示观察到该文档的事件（即该模型假定文档在搜索查询相关文档的过程中会被观察到）。观察到文档 d_j 给它的索引项所对应的随机变量赋予置信度。因此，一篇文档被观察到是该文档的索引项变量置信度上升的原因。索引项和文档变量被表示成网络中的结点。每条边从文档结点指向索引项结点，表示若文档被观察到，则会提高其索引项的置信度。

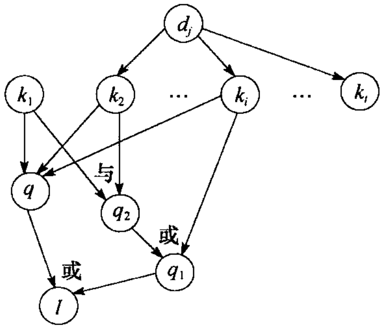


图 3-15 基本的推理网模型

用户查询的随机变量是用来表示这样的事件，即系统满足查询的信息需求。这个随机变量也由网络中的结点表示。在这个查询结点上的置信度是各查询项关联结点置信度的函数。因此，边由索引项指向了查询结点。图 3-15 说明了信息检索中的推理网。文档 d_j 有 k_2、k_i 和 k_t 作为其索
117 引项。这是通过把结点 d_j 指向结点 k_2、k_i 和 k_t 来建模的。查询 q 是由索引项 k_1、k_2 和 k_i 构成的。这是通过把结点 k_1、k_2 和 k_i 指向结点 q 来建模。注意图 3-15 也包含了三个额外的结点：q_2、q_1 和 I。结点 q_2 和 q_1 用来对查询 q 的替代布尔表达式 q_1 进行建模（在这种情况下，$q_1=(k_1 \wedge k_2) \vee k_i$）。当可获得这额外的信息时，用户的信息需求 I 可以同时由 q 和 q_1 支持。

接下来，我们会把注意力集中在由观察到文档 d_j 对于查询结点 q 的支持上。之后，我们讨论考虑同一信息需求 I 的不同的查询表式的影响。这是重要的，因为对于相同的信息需求，基于关键词的查询（例如 q）可以与类似于布尔表达式的查询表式（例如 q_1）结合，来产生更好的检索质量。虽然完整的推理网模型也包括文档结点和查询概念结点，但是上面讨论的模型总结了这种方法的本质。

可以使用一种简化的假设，即网络中所有的随机变量都是二值的。这似乎比较武断，但的确简化了建模任务，并且泛化能力强，可以获得信息检索问题中所有重要的关系。

定义 设 $\vec{k}$ 是一个 t 维向量，由 $\vec{k}=\{k_1, k_2, \cdots, k_t\}$ 定义，其中 $k_1, k_2, \cdots, k_t$ 是二值随机变量，即 $k_i \in \{0, 1\}$。这些变量定义了 $\vec{k}$ 的 2^t 种可能的状态。定义

$$on(i, \vec{k}) = \begin{cases} 1 & \text{如果根据} \vec{k}, k_i = 1 \\ 0 & \text{其他} \end{cases}$$

而且，设 d_j 是和文档 d_j 对应的二值随机变量，并设 q 是用户查询对应的二值随机变量。

118 注意 q 用来表示查询、相关的随机变量，以及网络中对应的结点。d_j 和索引项 k_i 也是同样的情况。我们允许这种在句法上的重载，因为我们指的是查询还是对应的随机变量应该是显而易见的。

对于查询 q，根据观察到文档 d_j 能提供多少证据来支持 q，对文档 d_j 进行排序。在推理网中，文档 d_j 的排序由 $P(q \wedge d_j)$ 计算，其中 q 和 d_j 是 $q=1$ 和 $d_j=1$ 对应的简短表示。总体来说，这样的排序是由如下给定：

$$\begin{aligned} P(q \wedge d_j) &= \sum_{\forall \vec{k}} P(q \wedge d_j \mid \vec{k}) \times P(\vec{k}) \\ &= \sum_{\forall \vec{k}} P(q \wedge d_j \wedge \vec{k}) \\ &= \sum_{\forall \vec{k}} P(q \mid d_j \wedge \vec{k}) \times P(d_j \wedge \vec{k}) \end{aligned}$$

$$= \sum_{\forall \vec{k}} P(q|\vec{k}) \times P(\vec{k}|d_j) \times P(d_j) \tag{3-58}$$

$$P(\overline{q \wedge d_j}) = 1 - P(q \wedge d_j)$$

这是由基本条件和贝叶斯定理获得的。注意 $P(q|d_j \wedge \vec{k}) = P(q|\vec{k})$，因为结点 k_i 把查询结点 q 和文档结点 d_j 分开。同样，记号 $\overline{q \wedge d_j}$ 是 $\neg(q \wedge d_j)$ 的简短表达。

文档结点 d_j 的实例化（即观察到的文档）分开了它的子结点，即索引项结点，使得它们相互独立（详见贝叶斯理论）。因此，给每个索引词结点 k_i 赋予的置信度可以分别计算。这表明 $P(\vec{k}|d_j)$ 可以按积的形式计算，于是从式（3-58）可得：

$$P(q \wedge d_j) = \sum_{\forall \vec{k}} P(q|\vec{k}) \times P(d_j) \times \left(\prod_{\forall i | on(i,\vec{k})=1} P(k_i|d_j) \times \prod_{\forall i | on(i,\vec{k})=0} P(\overline{k}_i|d_j) \right)$$

$$P(\overline{q \wedge d_j}) = 1 - P(q \wedge d_j) \tag{3-59}$$

其中 $P(\vec{k}|d_j) = 1 - P(k_i|d_j)$。通过对概率 $P(q|\vec{k})$、$P(k_i|d_j)$ 和 $P(d_j)$ 的适当定义，我们能使推理网广泛覆盖许多有价值的信息检索排序策略。接下来，我们将讨论如何使用推理网来推导布尔模型和 TF-IDF 排序公式。我们首先介绍概率 $P(d_j)$ 的定义。

（1）推理网的先验概率

由于文档结点是推理网的根结点，因此它们有一个由我们选择的先验概率分布。这个先验概率反映了观察到文档 d_j 这个事件的概率（为简化起见，假定一次观察单一的文档结点）。由于我们对任何文档并没有特别的先验倾向，因此我们通常采用均匀先验分布。例如， 119
在推理网最初的工作中［1610，1609］，观察到文档 d_j 的概率被设为 $1/N$，其中 N 是系统中文档的总数量。因此，

$$P(d_j) = \frac{1}{N} \text{ 和 } P(\overline{d}_j) = 1 - \frac{1}{N} \tag{3-60}$$

考虑到我们的文档集是由 N 篇文档组成的，为先验概率 $P(d_j)$ 选择的值为 $1/N$ 是简单自然的定义。然而，对于 $P(d_j)$ 的其他定义也是值得关注。例如，为了在模型中包含文档长度归一化，我们能得到如下的 $P(d_j)$。

$$P(d_j) = \frac{1}{|\vec{d}_j|} \text{ 和 } P(\overline{d}_j) = 1 - P(d_j)$$

其中 $|\vec{d}_j|$ 表示向量 $\vec{d}_j$ 的范数。因此，在这种情况下，文档向量的范数越大，其对应的先验概率就越小。这样的定义反映了我们对于向量排序策略的先验知识（这是在文档空间中归一化排序）。就像我们这里做的一样，具体应用环境下的先验知识应该在贝叶斯网的先验概率的定义中体现出来。

（2）布尔模型的推理网

这里我们说明如何从推理网推导出布尔模型。首先，对于布尔模型，先验概率由式（3-60）给出。对于条件概率 $P(k_i|d_j)$ 和 $P(q|\vec{k})$，定义如下：

$$P(k_i|d_j) = \begin{cases} 1 & \text{如果 } k_i \in d_j \\ 0 & \text{其他} \end{cases}$$

$$P(\overline{k}_i|d_j) = 1 - P(k_i|d_j)$$

这基本上表明，当文档 d_j 被观察到时，仅和文档 d_j 的索引项对应的结点是激活的（即有一个大于 0 的置信度）。例如，观察到文档结点 d_j，其索引项向量是恰好由索引项 k_2、k_i 和 k_t（见图 3-15）构成的，该向量激活了索引项结点 $\{k_2, k_i, k_t\}$ 而不激活其他结点。

当计算了索引项结点的置信度后，就能用它们来计算对于用户查询 q 的证据性支持，如下所示。

$$P(q\mid\vec{k})=\begin{cases}1 & \text{如果 } c(q)=c(\vec{k})\\ 0 & \text{其他}\end{cases}$$

$$P(\overline{q}\mid\vec{k})=1-P(q\mid\vec{k})$$

其中 $c(q)$ 和 $c(\vec{k})$ 是与 q 和 $\vec{k}$ 对应的合取分量，正如经典布尔模型中定义的那样。

把上述 $P(q\mid\vec{k})$、$P(k_i\mid d_j)$ 和 $P(d_j)$ 的定义代入式（3-59）中，可以容易地证明，检出的文档集合恰好是由 3.2.2 节定义的布尔模型返回的文档集合。因此，推理网可以轻松地
120 推导出布尔模型。

(3) 用于 TF-IDF 排序策略的推理网

文档结点的先验概率由式（3-60）给出。对于文档-索引项置信度，有：

$$P(k_i\mid d_j)=\alpha+(1-\alpha)\times\overline{f}_{i,j}\times\overline{IDF}_i$$

$$P(\overline{k}_i\mid d_j)=1-P(k_i\mid d_j) \tag{3-61}$$

其中 $\overline{f}_{i,j}$ 和 $\overline{IDF}_i$ 是归一化的项频和反比文档频率变量，并分别由如下给出：

$$\overline{f}_{i,j}=\frac{f_{i,j}}{\max_i f_{i,j}}$$

$$\overline{IDF}_i=\frac{\log\frac{N}{n_i}}{\log N}$$

这在［1611］的第 8 章中定义。参数 α 在 0～1 之间变换。经验证据表明 $\alpha=0.4$ 是一个良好的缺省值。

对于索引项-查询项置信度，有：

$$P(q\mid\vec{k})=\sum_{k_i\in q}\overline{f}_{i,j}\times w_q \tag{3-62}$$

$$P(\overline{q}\mid\vec{k})=1-P(q\mid\vec{k})$$

其中 w_q 是一个参数，用来在查询结点设置可达的最大置信度。一个合理的缺省值是 $w_q=1$。

把式（3-60）、式（3-61）和式（3-62）代入式（3-59），我们得到一个用于排序的 TF-IDF 式子。读者可参考［1611］获得在模型中定义条件概率的其他方法。

我们注意到用推理网计算的排序和用向量模型计算的排序不同。除了所生成的 TF-IDF 排序的特殊性外，推理网被证明在一般的文档集上能提供好的检索质量。其原因是推理网能使我们协调地结合不同证据源来提高最终的排序，我们接下来对此加以讨论。

(4) 结合证据源

在图 3-15 中，对于用户的信息需求 I，第一个查询结点 q 是基于关键词的标准查询表式，第二个查询 q_1 是类似于布尔表达式的查询表式（即一个由专家搜集的额外证据源）。这两个查询表式对于信息需求 I 共同的支持度可以用或操作（OR）来建模（即 $I=q\vee q_1$）。在这种情况下，由推理网提供的排序是如下计算的，

$$\begin{aligned}P(I\wedge d_j)&=\sum_{\vec{k}}P(I\mid\vec{k})\times P(\vec{k}\mid d_j)\times P(d_j)\\&=\sum_{\vec{k}}(1-P(\overline{q}\mid\vec{k})P(\overline{q}_1\mid\vec{k}))\times P(\vec{k}\mid d_j)\times P(d_j)\end{aligned} \tag{3-63}$$

121 相比各查询结点相互独立的情况，这可能产生更高的检索质量，如［1610］所示。

3. **信念网模型**

信念网模型（belief network model）[1352] 是推理网的一个变体，将文档和查询分开，其网络拓扑结构略微有所不同。

像推理网模型一样，在信念网中，所有的变量都是二值的。而且，每个索引项被看做是基本概念，所有索引项的集合被看做是概念空间。任何索引项的子集被解释为一个概念，可用于表示一篇文档或者用户查询。这种对称性导致了和推理网略有不同的网络拓扑，如图 3-16 所示。与在推理网模型中那样，查询 q 被建模为二值随机变量，指向这个查询的是组成它的概念索引项结点。

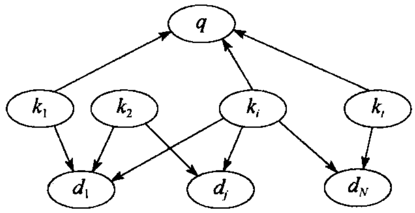


图 3-16 基本信念网模型

文档的处理类似于用户查询（即两者都是概念空间上的概念）。因此，与推理网模型不同，文档结点被组成文档的索引项结点所指向。

与给定查询 q 相关的文档 d_j 的排序被解释为概念匹配关系，反映了概念 d_j 被概念 q 覆盖的程度。

假设

在信念网模型中，采用 $P(d_j|q)$ 作为文档 d_j 对于查询 q 的排序。

通过应用贝叶斯定理，能得到 $P(d_j|q)=P(d_j\wedge q)/P(q)$。因为 $P(q)$ 对于文档集中所有的文档都是常数，因此得到 $P(d_j|q)\sim P(d_j\wedge q)$，即赋给文档 d_j 的排序直接正比于 $P(d_j\wedge q)$。因此，

$$P(d_j|q) \sim \sum_{\forall \vec{k}} P(d_j \wedge q|\vec{k}) \times P(\vec{k})$$

在图 3-16 的信念网中，对索引项变量的实例化在逻辑上分离了结点 q 和 d，使得它们相互独立（即信念网的文档和查询部分在逻辑上被索引项结点的实例分离）。因此， 122

$$P(d_j|q) \sim \sum_{\forall \vec{k}} P(d_j|\vec{k}) \times P(q|\vec{k}) \times P(\vec{k}) \tag{3-64}$$

为了实现推理网，需要定义条件概率 $P(q|\vec{k})$ 和 $P(d_j|\vec{k})$。对这些概率的不同定义允许对不同的排序策略建模（对应于不同的信息检索模型）。例如，对于向量模型，概率 $P(q|\vec{k})$ 和 $P(d_j|\vec{k})$ 按如下方式定义。定义向量 $\vec{k}_i$：

$$\vec{k}_i = \vec{k}|on(i,\vec{k}) = 1 \wedge \forall_{j\neq i} on(i,\vec{k}) = 0$$

向量 $\vec{k}_i$ 表示当结点 k_i 是激活的，而所有其他结点都不激活的情况下 $\vec{k}$ 的状态。这么做是因为 TF-IDF 排序策略需要累加各个索引项的贡献，而 $\vec{k}_i$ 使我们能单独考虑索引项 k_i 的贡献。接下来有：

$$P(q|\vec{k}) = \begin{cases} \dfrac{w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,q}^2}} & \text{如果 } \vec{k} = \vec{k}_i \wedge on(i,\vec{q}) = 1 \\ 0 & \text{其他} \end{cases}$$

$$P(\overline{q}|\vec{k}) = 1 - P(q|\vec{k})$$

而且，定义：

$$P(d_j|\vec{k}) = \begin{cases} \dfrac{w_{i,j}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2}} & \text{如果 } \vec{k} = \vec{k}_i \wedge on(i,\vec{d}_j) = 1 \\ 0 & \text{其他} \end{cases}$$

$$P(\overline{d_j} \mid \vec{k}) = 1 - P(d_j \mid \vec{k})$$

那么，由 $P(d_j|q)$ 定义的检出文档的排序和由式（3-10）用于向量模型的排序就恰好相同了。

4. 贝叶斯网的计算代价

在推理网模型中，根据式（3-58），只考察含有激活文档结点的状态。因此，计算排序的代价与文档集中的文档数成线性关系。对于传统的文档集，倒排表等索引结构（见第9章）可把排序计算限制在那些和查询有共同索引项的文档上。因此，计算推理网排序的代价和计算向量模型排序的代价有着相同的复杂度。信念网模型的情况也是这样。

因此，这里讨论的贝叶斯网模型在排序计算上没有附加明显的额外代价。这是因为这里
123 呈现的网络不包含环，而这意味着完成信念传播的时间是与网络中结点的数量成正比的。

3.6 其他模型

本节讨论的文档检索模型和排序方法不是直接由经典的信息检索模型衍化而来的。这些模型覆盖超文本、Web 排序、结构化文本和多媒体。我们先在这里加以简介，然后在后面的各章中更详细地讨论。

3.6.1 超文本模型

超文本为超文本标记语言（Hypertext Markup Language，HTML）和超文本传输协议（Hypertext Transfer Protocol，HTTP）提供了设计基础，是 Web 的起源。

我们接下来回顾超文本模型背后的一些概念，解释它是如何与如今的 Web 联系起来的。

一个和文字书写相关的基本概念是*序列*（sequencing）。书面文本通常被看做是序列化阅读。读者无法通过随机阅读文章的片段来理解作者要表达的信息。人们可通过文章的结构来跳过其中的一些部分，但这可能导致读者和作者之间信息的错误传达。可见，在大部分书面文章里隐含着序列化的组织结构。当读者无法认识并依循这个结构时，他们经常无法捕捉到作者所要表达的信息的本质。

然而，有的时候，我们寻找的信息是包含在整个正文中的，不能轻易地通过顺序阅读来捕捉到。例如，当浏览一本关于人类战争历史的书时，我们可能暂时对欧洲的局部战争感兴趣。我们知道这个信息是在这本书中，但可能不容易找到它，因为作者不是按照这样的目的来组织他的作品的（他可能按照编年史来组织战争）。在这样的情况下，需要对正文有不同的组织结构。由于不能重写全文，解决的方法是除了已经存在的结构之外，再定义一个新的组织结构。实现该目的的一种方法是通过超文本的设计。

*超文本*是一种高层次的、交互式的导航结构，允许在计算机屏幕上非连续地浏览文本。它基本上由结点构成，这些结点在图结构中由有向边连接。每个结点表示正文的一个区域，可能是书中的一章、文中的一节，或者一个网页。两个结点 A 和 B 可能通过有向边 l_{AB} 连接，这使这两个结点所代表的文本关联起来。在这种情况下，读者可能会在阅读 A 结点相关的正文时转移到 B 结点。

超文本链接 l_{AB} 最传统的形式是附着在结点 A 正文内特定的字符串上。这样的字符串被特别标记出来（例如，其字符可能有不同颜色或者有下划线），表示有潜在的链接。当阅读文章时，用户可能遇到这个有标记的字符串。如果用户点击这个字符串，追踪其潜在的有向链接，一个和目标结点关联的新文本区域就在屏幕上显示出来了。

导航超文本的过程可以被理解为遍历有向图，其结点表示文本结点。而遍历这个图，读

者会看到由超文本设计人员所构想的信息流。例如，重新考察我们之前关于人类战争的书的例子。人们可能设计一个由两个不同的 Web 构成的超文本（这里，Web 指的只是一个由超文本中所有链接的某个子集形成的连通分支）。而第一个 Web 可能被设计为按照编年史组织的欧洲局部战争，第二个 Web 可能被设计为按照国家组织的欧洲局部战争。这样，用户能获得这个超文本按照不同的组织结构提供的信息。 124

当超文本非常大时，用户可能在超文本的组织结构中迷失。其效果是用户可能开始做出糟糕的浏览决定，这可能转移他原来的主要目标（通常是找到超文本中的某条信息）。当这种情况发生时，用户被认为是*迷失在超空间中*（lost in hyperspace）[836]。为了避免这个问题，需要在超文本中包含一份超文本映射图（hypertext map），显示用户在哪里。这个映射图最简单的形式是一个显示了当前被访问的结点的有向图。这张图也可以包含用户至今已访问过的路径信息。当用户访问已经访问过的路径时，可以用这张图来提醒用户。

当导航一篇超文本时，用户限制在由超文本设计人员力图构想出来的信息流中。因此，设计超文本的任务需要考虑其潜在用户的需求。这表明在开始实际实现这个超文本前，需要一个需求分析阶段。这样的需求分析是非常重要的，但是经常被忽视。

而且，在超文本导航阶段，用户可能觉得难于自行导引。这种困难甚至发生在有导航工具的情况下，例如我们讨论的超文本映射图。一种可能的原因是超文本组织过于复杂，含有过多的链接。为了避免这个问题，超文本应该有一个更简单的结构，可以在任何时候被用户快速地记住。例如，超文本可能按照层次化的结构组织起来，使得导航任务变得简单。

超文本结构的定义应该在领域建模阶段完成（在需求分析阶段之后）。在对领域建模之后，用户的界面设计应该在实现前先总结出来。直到那时，我们才能说我们有了一个为眼下的应用设计合适的超文本结构。然而，在 Web 中，通常是在不关心需求分析、领域建模和用户界面设计的情况下做出网页来的。因此，网页经常设计得很糟糕，不能提供用户一个合适的超文本结构来支持信息搜索任务。

对于大的超文本，用户可能难于在整个图中定位感兴趣的部分。为了推动这个初始的定位步骤，可以使用基于索引项的搜索。在［1708］中，Manber 讨论了这种方法的优点。

3.6.2 基于 Web 的模型

第一个 Web 搜索引擎本质上是信息检索引擎，其排序是基于我们这里讨论的信息检索模型。主要的区别是：1）文档集是由网页（而不是文档）组成的；2）要先爬取网页；3）文档集要大得多。第三个区别，即大量的网页，也意味着仅依靠正文排序方法不会像从前对小规模文档集那样有效。每个查询词检出了太多的文档，导致了单一用户查询的结果包含了数以千计的文档，大部分的文档和用户不相关。因此，这些引擎产生的检索结果经常不令人满意。 125

基本的信息检索引擎缺少了一项关键的革新——利用网页中包含的链接信息来修改排序。有两个基本的方法，即 PageRank[263] 和 Hubs & Authorities[911]。由于他们提供了 Web 中的特殊排序方法，因此我们将在第 11 章中讨论。结合了正文（基于向量的排序）和链接信息的排序方法的例子可见［1478］。

3.6.3 结构化文本检索

在本章中讨论的所有信息检索模型都把正文看做是没有特殊结构的字符串。也就是说，在节、子节、章和图中的信息没有被包含在模型中，也没有用于排序。然而，结构上的信息

可能对用户的特殊搜索是重要的。举例来说，某个用户需要检索一本书，却忘了书名和作者。但是，用户记得这本书包含了埃菲尔铁塔的图片，其小节的题目含有词汇“France”。在这种情况下，定义查询“France”是没有帮助的，因为这会返回太多的文档。用户有他感兴趣的那本书足够多的信息，但是无法用我们已经定义的信息检索模型来定义他的查询。

这个问题的解决方法是采用文档的文本结构来提高检索。为了效率的原因，这意味着要建立特殊的索引结构，使得其更适合对结构信息编码，我们将在第 13 章中讨论。

3.6.4 多媒体检索

多媒体数据，即图像、音频和视频，经常缺乏对应的文本，使得它们的检索更具有挑战性。尤其是，我们这里讨论的信息检索模型在多媒体数据上收效甚微。需要采用的检索策略和排序函数与文本检索的策略很不一样。而且，甚至查询定义也是不同的。

除了这些特殊性外，多媒体数据是 Web 的一个主要部分。Web 确实是一种天生的多媒体媒介。因此，我们对多媒体的检索方法和技术很感兴趣，并在第 14 章非常详细地讨论。

3.6.5 企业和垂直搜索

企业搜索是在公司文档集上搜索感兴趣信息的任务。虽然大部分的信息是由公司的文档集提供，但也需要用 Web 中的信息来补充。许多没有出现在 Web 中的问题，例如隐私性、拥有权和许可，在企业搜索中是重要的。在第 15 章中，我们详细地讨论在部署最先进的企
126 业搜索方法的过程中遇到的挑战。

垂直文档集是含有某给定领域知识的专业文档的文档库。举例来说，Lexis-Nexis 提供了专注于两个主要垂直领域的全文搜索，一个在商业领域，另一个在法律领域。同样，Medline 提供了生命科学领域，尤其是生物医药方面的垂直文档集上的搜索。更多的关于这些系统的详情见第 16 章。

垂直文档集展现了关于搜索和检索方面特定的挑战。因此，有可能需要利用领域相关的知识来提高结果——这个问题我们这里不讨论。为了涵盖信息检索在健康方面垂直文档集的应用，有兴趣的读者可以参考［539，588，480］和 W. Hersh 关于这个话题的扩展读物［750，753，751，752，749］。为了涵盖信息检索在法律方面垂直文档集的应用，读者可以参考［204，481，482，638，692，1381］。

3.7 趋势和研究问题

主要有三类产品和系统能直接从信息检索模型的研究中获益：图书馆系统、专用检索系统和 Web。

对于图书馆系统，目前有大量的兴趣关注认知行为问题，尤其是更好地理解用户采用哪种标准来判断相关性。从计算机科学家的视角看，主要的问题是关于用户行为的知识如何影响排序策略和系统实现的用户界面。一个相关的问题是调查如何整合标准的商业图书馆系统，其中许多是基于表单和布尔检索，以及 Web 中的多媒体库。考虑到如今的学生有网上的搜索体验，他们在图书馆中不希望遇到陈旧的图书馆系统，我们显然需要更复杂的搜索功能。

专用检索系统是为了设想的特殊需求开发的。例如，第 16 章讨论的 Lexis-Nexis 检索系统，提供了获取大量法律和商业文档集的渠道，是垂直信息检索系统的一个好例子。在这样的系统中，主要的问题是如何检索可能和用户的信息需求相关的（几乎）所有文档，而不检

出大量不相关的文档。在这种情况下，非常需要复杂的排序算法。由于基于单一的证据源的排序不可能提供合适的答案，因此结合多个证据源，尤其是领域相关的专业知识的方法是合适的。在这样的方向上，开发企业分类体系是一个常用的方法，即使这种基于分类体系的搜索策略需要仔细地调整才能产生相关的结果。更多关于企业搜索系统的趋势和研究问题，可见第 15 章。

在 Web 中，情况是相当不同和独特的，正如我们在第 11 章中看到的。实际上，Web 用户经常不知道他要什么或者很难适当组织其需求。因此，关于高级用户界面的研究是一个热门的话题。从排序引擎的观点看，一个有趣的问题是研究用户界面的某个具体范式是如何影响排序的。同样，使用用户的偏好信息，通常称为个性化（personalization），继续值得大量 127
关注。一种探索用户偏好的形式是在结果页面中检查用户点击的模式［17，841，844］，但还需要设想其他的方法。

3.8 文献讨论

一份在索引项权重方面非常早期的工作，是 1957 年 Luhn 完成的。他假设文档中索引项的权重和该文档内的项频成正比［1062］。早在 1960 年，Maron 和 Kuhns［1093］就讨论了信息检索中相关性和概率索引的问题。在反比文档频率方面开创性的工作是 Sparck Jones［1504］。紧接着，Salton、Yang 和 Wong 合作，提出了项频和反比文档频率的结合，获得了现在公认的经典向量空间模型［1418，1416］。不久之后的 1976 年，Robertson 和 Sparck Jones 提出现在的经典概率模型。

7 年之后，Salton 和 McGill 写了一本书［1414］，成为该领域的标准。这本书彻底覆盖了信息检索里的三个经典模型，也就是布尔、向量和概率模型。另一个里程碑是 van Rijsbergen［1624］的书，除了覆盖了三个经典模型外，还提供了一份关于概率模型彻底的、令人欣赏的讨论。由 Frakes 和 Baeza-Yates［582］编辑的书展示了信息检索的多个数据结构和算法，并更贴近当下。而且，他包含了关于 Harman［701］的排序算法的讨论，提供了关于从 1960—1990 年的信息检索历史的有趣观点。

布尔运算及其实现在［1667］中涵盖。用于信息检索的布尔查询的不充分性早在 Verhoeff、Goffman 和 Belzer［1636］中就描述了。把布尔表式迁移到其他框架下的问题获得了大量的关注。Bookstein 讨论了把布尔和带权重的检索系统［227］合并起来的问题，以及在概率检索中实现布尔结构［229］。Losee 和 Bookstein［1050］把布尔查询的使用和概率检索结合起来。Anick 等人［60］提出了基于自然语言的布尔检索的界面。基于词典的布尔检索系统在［994］中提出。

向量模型可能是在信息检索研究领域中最流行的模型。它的流行很大程度上要归功于 Salton 及其合作者［1413，1418］的长期研究。这项研究的大部分以 Cornell 大学开发的 SMART 检索系统［1408，1410，1756］为中心。向量模型中索引项的权重已经被彻底研究。简单的索引项权重早期由 Salton 和 Lesk［1413］使用。Sparck Jones 介绍了 IDF 因子［1504，1505］，Salton 和 Yang 证明了其在提高检索上的效果［1418］。Yu 和 Salton［1756］进一步研究了索引项权重在最终排序中的效果。Salton 和 Buckley［1410］总结了 20 年来在 SMART 系统上索引项权重方面的实验。Raghavan 和 Wong［1327］提供了对向量模型的严谨分析。Singhal、Buckley 和 Mitra［1484］讨论了主轴文档长度归一化，这是一项修改归一化因子以提高在不同文档集上结果的技术。 128

概率模型由 Robertson 和 Sparck Jones［1365］介绍的，在［1624］中进行了彻底讨论。

关于该模型的实验性的研究是由 Sparck Jones[1506，1507] 进行的，使用从用户得来的反馈信息来估计初始概率。Croft 和 Harper[452] 提出了不使用用户的反馈信息估计这些概率的方法。Croft[450] 后来在模型中增加了文档内频率权重。Fuhr 通过多项式检索函数讨论了概率索引 [597，598]。Cooper、Gey 和 Dabney[423] 和之后的 Gey[622] 提出在概率检索中使用 logistic 回归。Lee 和 Kantor[993] 研究了不一致的专家判断在概率检索中的效果。Fuhr[599] 回顾了经典概率模型中的不同变体。Cooper[422] 在一份开创性的论文中，提出了在信息检索中使用概率排序原则的麻烦。最近的关于概率模型的综述是 Robertson 和 Zaragoza[1369]。

本书涵盖的（用于信息检索的）模糊集模型是由 Ogawa、Morita 和 Kobayashi 提出的 [1224]。在信息检索中使用模糊理论可以追溯到 20 世纪 70 年代 Radecki[1314，1315，1316，1317]、Sachs[1403] 和 Tahani[1553] 的工作。Bookstein[228] 提出利用模糊运算符处理带权重的布尔搜索。Kraft 和 Buel 利用模糊集来泛化布尔系统 [938]。Miyamoto、Kraft 和 Nakayama[1143] 讨论了使用共现和模糊操作产生的伪同义词典。后来，Miyamoto 和 Nakayama[1144] 讨论了在信息检索中使用这个词典的情况。有些相关的内容是将模糊论应用到数据库系统中对近似答案进行排序 [1353]。

扩展布尔模型是由 Salton、Fox 和 Wu[1412] 介绍的。Lee、Kim、Kim 和 Lee[996] 讨论了在扩展布尔模型上布尔运算的评价，而该模型的性质在 [995] 中讨论了。基于集合的模型是由 Pôssas、Ziviani、Meira 和 Ribeiro-Neto[1294] 介绍的，这是作为一种使用索引项之间的相关性提升结果的方法。关于查询处理的模型扩展在 [1293，1296] 中。一个对于模型及其扩展更彻底的评价可以在 [1295] 中找到，我们用来作为这里讨论的基础。

广义向量空间模型是于 1985 年由 Wong、Ziarko 和 Wong[1718，1717] 介绍的。潜在语义索引于 1988 年由 Furnas 等人 [614] 引入。在后来的论文中，Bartell、Cottrell 和 Belew[153] 证明潜在语义索引能被解释为多维尺度变化的特例。

对于信息检索中的神经网络模型，本书中的讨论主要基于 Wilkinson 和 Hingston[1697] 的工作。但是，我们也能从 Kwok 在讨论该项内容及其相关主题的著作 [949，950，951，952] 中获益。

BM25 模型是一系列旨在提高 Okapi 系统性能的实验所获得的成果 [1366，1367，1368]，当时 Okapi 系统参与了 TREC 会议（详细见第 4 章关于 TREC 会议的内容）。更多关于 BM25 的信息可见 [1369]。推理网模型是由 Turtle 和 Croft[1609，1610] 于 1990 年介绍的。Haines 和 Croft[693] 讨论了用于用户相关反馈的推理网。Callan、Lu 和 Croft [323] 使用了推理网来搜索分布式文档集。Callan[319] 在他的论文中讨论了推理网在信息过滤中的应用。信念网模型是更易掌握的推理网变体，由 Ribeiro-Neto 和 Muntz[1352] 提
129 出。信息检索中的贝叶斯网的回顾可见 [445]。

语言模型最初由 Kalt[864] 提出，接着是 Ponte 和 Croft[1270]，他们的工作被看做是把语言模型应用到信息检索中的里程碑。许多研究紧随其后，例如 Berger 和 Lafferty[187]，Miller、Leek 和 Schwartz[1132]，Hiemstra 和 Kraaij[762]。研究人员提出了许多平滑方法，大部分在语音识别的领域里。在本章中，我们使用的平滑的形式是 Chen 和 Goodman [366]。我们对于平滑的讨论是基于 Zhai 和 Lafferty[1772] 的工作。信息检索中语言模型的完整论述最近由 Zhai[1770，1771] 发表。

有些基于类似于语言模型方法的相关模型，是由 Amati 和 Rijsbergen[39] 提出的随机差异模型框架（Divergence Form Randomness，DFR）。随机差异模型基于这样的想法，即

文档内索引项的频率与文档集内的频率的差异越大，这个项携带的信息越多。这意味着索引项权重应该是与文档集内项频的概率成反比，其中后者的概率是由模型的随机性获得。用于对随机性建模的分布，包括二项分布、Boise-Einstein 分布，以及 Boise-Einstein 的几何近似。关于随机差异更详细的讨论，读者可参考 Amati 的博士论文 [38]。

已经发现信息检索模型可以用于文本集之外的领域，例如多媒体、定向广告和数据库。对于信息检索在多媒体中的应用见第 14 章。对于信息检索在定向广告中的应用见 [270，275，277，954，1319，1350]。对于信息检索模型在数据库和 Web 数据库中的应用可见 [314，315，317，471，641，1639]。

Web 是一个松散的超文本，阅读一些关于这个主题的文献是有用的。一个关于超文本的经典参考是 Nielson [836] 的书。另一本流行的参考书是 Shneiderman 和 Kearsley[1470] 的书。Conklin[412] 给出了该领域的介绍性综述。《Communications of the ACM》贡献了一期关于超媒体的专辑 [410]，其中详细讨论了 Dexter 模型——一个关于基本超媒体概念术语和语义的参考标准。后续的版本 [411] 专门用来描述各种不同的模型，这些模型支持超媒体应用的设计。对于 Web 及其技术的专门参考文献可见第 11 章。 130

第4章

Modern Information Retrieval: The Concepts and Technology behind Search, 2E

检索评价

4.1 介绍

评价信息检索系统就是度量系统能在多大程度上满足用户的信息需求。这自然是困难的，尤其是考虑到不同的用户对于相同的结果集可能会有不同的解释。虽然如此，我们依然可以定义一个近似的指标，平均而言，和用户的偏好有紧密的相关性。在本章中，我们将讨论这些指标和它们的应用。

没有合适的评价方法，我们就无法确定信息检索系统能运转得多好，也就不能把它的检索质量和别的信息检索系统进行客观的比较。因此，信息检索系统的系统化评价应能够回答在其实际日常运行中产生的问题，例如：

1）若提出了一种对排序函数的修改方法，我们是否应该开始启用它？

2）若构想出一个新的概率排序函数，它是否比向量模型和 BM25 排序得更好？

3）对于 Web 查询，如商业查询、产品查询，还是地理查询，哪种给定的排序修改方法最有效？

缺乏适当的评价就无法客观地回答这些问题，也无法对排序函数进行良好的调整。

> **检索评价**针对信息检索系统响应用户查询的返回结果，系统化地给出了一个量
> 化的指标。这个指标应该和检索结果与用户的相关性直接联系。计算这个指标的通
> 131 常方法是，对于给定的一组查询，比较由系统产生的结果和由人产生的结果。

注意到这里的检索评价意味着评价结果的质量，而不是系统的性能（即它能多快地处理查询）。因此，我们避免使用*检索性能评价*（retrieval performance evaluation）这样的术语，但这样的术语却经常在专业文献中使用，用来表示上面定义的检索评价。

我们的定义包含了问题的一方面，即结果质量的评价，但不包含那些会影响用户判断方面的因素。举个例子，在用户体验方面的一个主要影响因素是用户界面（User Interface, UI），但是我们上面的定义并不包括界面的特性，例如布局、颜色、图标和时延。它也不考虑结果的相关性会受到诸多因素的影响，例如查询提交时的背景情况、用户的偏好、提交的时间等。而且，查询可能是一项复杂的检索任务中的一部分，其目的只是满足一个有意义的信息需求。这些通常更难以有效地度量和评价，需要耗费大量的时间来营造适当的评价环境。

尽管有这些缺点，但对于查询结果赋予一个数值指标的评价过程还是被广泛地采用了。可能是因为这更简单，并能以相对低廉的代价重复多次。可重复性是其主要的优点，因为这允许在相对短时间内研究更大批的查询和它们的结果。这是至关重要的，因为这使我们能够通过仔细检查指标是如何因排序函数的改变而受影响的，从而深入了解在排序函数中哪些因素是不起作用的。

在本章中，我们将讨论信息检索系统的检索评价。由于评价通常基于一个测试参考集，所以在本章中我们会讨论不同的文档集。我们的讨论首先是涵盖了 Cyril Cleverdon[395, 399, 398] 的原创性工作，其到达了 Cranfield 范式的顶峰——信息检索评价指标发展的基础。Cranfield 框架之外的评测方法将在 4.5.5 节讨论。

4.2 Cranfield范式

信息检索系统的系统化评价是20世纪50年代由Cyril Cleverdon开创的早期实验结果所建立的，这在所谓的Cranfield实验中达到了顶峰。这些实验提供了对信息检索系统进行评价的基础，即我们现在要讨论的内容。

4.2.1 历史简述

回到1952年，英国Cranfield Aeronautics学院的图书馆员Cyril Cleverdon，注意到一个由美国政府图书馆员Mortimer Taube提出的新的索引系统，称为“Uniterm系统”（Uniterm System）。Taube仔细地分析了近40 000个主题词（subject headings），发现它们仅由7000个不同的词组成。然后，他提出对文档的索引应该仅基于不同词的集合，这就是名字Uniterm的由来。Cleverdon为此所吸引，便和同事Bob Thorne做了一个小实验。他用Uniterm手工地对200篇文档进行索引，并要求Thorne运行一些查询。这个简单的实验使Cleverdon走上了依靠实验来评价索引系统的终生道路，其最终的顶峰是精度和召回率等现代化指标，如今这些指标在信息检索中是非常流行的。 132

新的Uniterm系统让那些拥护更复杂的索引系统的人感到困扰，比方说主题词，它在当时被大部分卡片目录系统所使用。Uniterm系统似乎过分简单了，并缺少主题词的语义性。虽然争论很激烈，但是没有可靠且可用的数据来对这两个索引系统进行直接的比较。对Cleverdon来说，这两种索引系统显然需要一种独立的评价方法，他将自己投入到这项任务中去。

Cleverdon获得了美国国家科学基金会（National Science Foundation，NSF）的资助，对4个不同的索引系统进行比较，其中包括Uniterm系统。这个项目，被称为Cranfield-1，需要用各种索引方法对18 000篇关于宇航工程的论文手工建立索引，并对1200篇搜索问题的结果进行评价。每个问题来自一篇单一的文档，当那篇文档被待测试的索引系统正确编目时，搜索被认为是成功的。结果显示，4个索引系统对于结果的精度而言基本上是相同的。

除了结论本身外，这些大量的、辛苦的实验也提供了有趣的见解。例如，如果不附上精度（即检出相关文档的比例）的信息，召回率（即相关文档被检出的比例）的价值就不大了。验证实验表明，在召回率和精度之间有反比的关系。这清晰地说明了，在任何某个具体的索引系统中，不可能同时提升精度和召回率。

下一步是要设计一组实验，对每一个索引系统单独进行更为彻底的评价。对每个搜索问题，检查数据集中所有的文档，并判断它们和这个问题的相关性。这显然意味着用于评价的数据集的规模要小，否则评价过程的代价将变得非常高。实验设定用了1400篇文档和279个问题。6名学生花了3个月的时间比较每篇文档和每个问题，并决定文档是否和这个问题相关。其结果是一个由文档、查询和对每个“文档-查询”对的相关性评价组成的参考测试集，称为Cranfield-2数据集。实验的主要结果是，精度-召回率曲线呈现出大家所熟知的双曲线形状。

在Cranfield-2实验中也观察到，在实际情况中，大多数搜索不需要高的召回率。相反，大多数用户只是要求一些相关的答案——这个结论在Web中更有效。

Cranfield-2实验建立了信息检索中的现代实验法的基础。通过和专家产生的相关性评价进行比较，相同的文档和查询被用来评价不同的排序系统。而且，精度和召回率如今是评价排序质量的必选指标。在产生了相关性评价之后，这种设定的统一性使得可以进行快速的评价，这使得该方法具有巨大的实用价值。

133 Cranfield-2 范式的主要缺点是其潜在的简化假设。用户的信息需求被假设是静态的，文档的相关性被假设是和其他文档的相关性无关的。而且，它假定一组单一的相关性评价结果反映了用户群的观点，同时假定对任何给定的查询所有的相关文档是已知的。通过查看如今的 Web 用户，我们观察到所有这些假设都不成立。尽管有这些局限，由 Cranfield-2 实验建立的评测过程如今仍然被广泛采用，这主要是因为它对不同信息检索系统的结果提供了客观、易于解释且可比较的指标。

4.2.2 参考集

参考集允许对不同排序函数产生的结果进行直接的比较。它们基于 Cranfield 实验定义如下：

> **参考集**：参考集是由一组预先选择的文档集$\mathcal{D}$、一组用于测试的信息需求描述 I 和一组与每个二元组 $[i_m, d_j]$ 对应的二元相关性评价组成的，其中 $i_m \in I$ 且 $d_j \in \mathcal{D}$。如果文档 d_j 与信息需求 i_m 是不相关的，那么相关性评价值为 0；如果 d_j 和 i_m 是相关的，其值为 1。

相关性评价是由专家产生的，理想情况下应该为每一个信息需求-文档二元组提供相关性判断。显然，这仅仅对于小的文档集是可行的，正如 Cranfield 实验中那样。并且需要注意的是，相关性评价是针对信息需求描述，而不是针对查询。这是因为把信息需求描述翻译为查询也被认为是评价过程的一部分，需要由被评价的检索算法完成。对于 Web 而言显然不是这样，用户需要直接指定查询。参考集具有许多重要的优点，具体如下所示：

- 给定参考集，信息检索系统的评价可以迅速完成，这使得我们可以对不同系统、不同排序函数进行比较。
- 出于验证目的，系统可以在评价之后重新进行评价，即参考集提供了实验的可重复性。
- 可以针对特殊种类的信息需求建立不同的参考集，这使我们能够对排序函数的性质有更加深入的理解。

由于这些优点，参考集继续广泛用于评价信息检索系统，更详细的内容见 4.4 节。

4.3 检索指标

本节将回顾评价信息检索系统的检索质量（即结果质量）的不同指标。其中，最广泛使

134 用的是召回率和精度。

4.3.1 精度和召回率

考察（一个测试参考集的）信息需求 I 和相关文档集合 R。设 $|R|$ 是这个集合内文档的数目。假设某个给定的（待评测的）检索算法处理了这个信息需求 I，生成了一组答案 A。设 $|A|$ 是这个集合中文档的数目。而且，设 $|R \cap A|$ 是集合 R 和 A 交集中的文档的数目，如图 4-1 所示。那么，精度和召回率的指标定义如下：

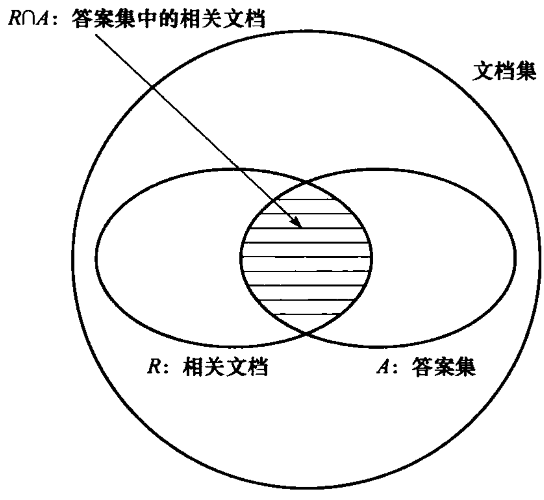


图 4-1 对于给定信息需求 I 的精度和召回率

- **精度**（Precision）是检出文档（集合 A）中相关文档的比例，即

$$精度 = p = \frac{|R \cap A|}{|A|} \tag{4-1}$$

- **召回率**（Recall）是相关文档（集合 R）被检出的比例，即

$$召回率 = r = \frac{|R \cap A|}{|R|} \tag{4-2}$$

上面定义的精度和召回率假设答案集 A 中所有的文档已经被检查过（或者见过）。然而，用户通常不会马上见到答案集 A 中所有的文档。相反，A 中的文档首先根据分数排序。然后，用户从第一篇文档开始检查这个排序队列。在这种情况下，精度和召回率的值会随着用户对答案集 A 的检查而变化。因此，需要按如下方法绘制精度-召回率曲线。 135

假设一个参考集和一组测试查询，并假设对于给定的查询 q_1 具有相关文档集合 R_1，该集合是一组专家所确定的。不失一般性，再假设集合 R_1 由如下的文档组成：

$$R_1 = \{d_3, d_5, d_9, d_{25}, d_{39}, d_{44}, d_{56}, d_{71}, d_{89}, d_{123}\}$$

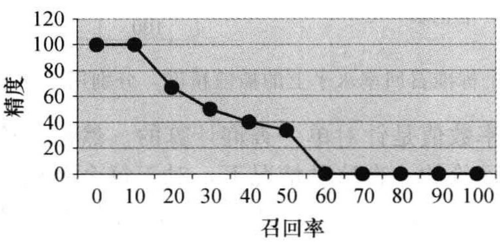
这样，根据专家评估，有 10 篇文档和查询 q_1 相关。

现在考虑一个刚刚设计的新检索系统。假设这个算法对于查询 q_1，在答案集中返回了一列排过序的文档，如下所示。

对应查询 q_1 的排序：

1. d_{123} •	6. d_9 •	11. d_{38}
2. d_{84}	7. d_{511}	12. d_{48}
3. d_{56} •	8. d_{129}	13. d_{250}
4. d_6	9. d_{187}	14. d_{113}
5. d_8	10. d_{25} •	15. d_3 •

和查询 q_1 相关的文档，即那些属于集合 R_1 的文档，在其编号后面加注了一个小圆点。如果我们从第一篇文档开始，检查这个排序，可以观察到如下的情况。第一，排在第一位的文档 d_{123} 是相关的，而且这个文档占到了 R_1 中所有相关文档的 10%。这样，我们说在 10% 的召回率上有 100% 的精度。第二，文档 d_{56} 排在第三位，是下一篇相关文档。此时，我们说在 20% 的召回率（10 篇相关文档中找出了 2 篇）上有 66.6% 的精度（3 篇中 2 篇是相关的）。第三，如果我们接着检查生成的排序队列，就能绘制一条精度-召回率的曲线，如图 4-2 所示。召回率高于 50% 后，精度下降到了接近于 0，因为不是所有的相关文档都被检出。这个精度-召回率的曲线通常基于 11（而不是 10）个标准召回率水平，分别是 0%、10%、20%、…、100%。对于召回率水平为 0% 的情况，其精度是通过我们接下来讨论的插值方式获得的。



召回率	精度
0	100
10	100
20	66.6
30	50
40	40
50	33.3
60	0
70	0
80	0
90	0
100	0

图 4-2　对于查询 q_1 在 11 个标准召回率水平上的精度，分别用图和表显示 136

现在考虑第二个不同的查询 q_2，根据专家判断，其相关文档集合如下：

$$R_2 = \{d_3, d_{56}, d_{129}\}$$

而且，假设使用和我们考察 q_1 时相同的检索算法来处理查询 q_2，并返回如下所示的排过序的结果队列。

对应查询 q_2 的排序：

1. d_{425}	6. d_{615}	11. d_{193}
2. d_{87}	7. d_{512}	12. d_{715}
3. d_{56} •	8. d_{129} •	13. d_{810}
4. d_{32}	9. d_4	14. d_5
5. d_{124}	10. d_{130}	15. d_3 •

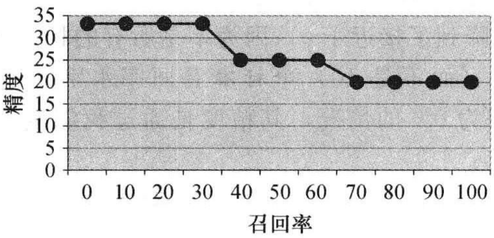
与前面的情况类似，其中的圆点标记了相关文档。在这个例子中，排序列表中第一个相关文档是 d_{56}，其召回率水平是 33.3%(精度也是 33.3%)，因为，此时所有相关文档的 1/3 已经被观察到了。第二个相关文档是 d_{129}，其召回率水平是 66.6%(其精度为 25%)。第三个相关文档是 d_3，召回率是 100%(其精度是 20%)。在 11 个标准召回率水平上的精度按如下方法进行插值。

设 r_j，$j \in \{0, 1, 2, \cdots, 10\}$，是第 j 个标准召回率水平（即 r_5 表示召回率水平 50%）。那么，

$$P(r_j) = \max_{\forall r | r_j \leq r} P(r) \tag{4-3}$$

这说明了在第 j 个标准召回率水平的插值精度是所有高于 r_j 的召回率水平所对应的精度当中的最大值。

在我们上一个例子中，这个插值规则产生的精度和召回率数值在图 4-3 中展示。在召回率水平 0%、10%、20% 和 30% 上，插值后的精度等于 33.3%，对应于召回率水平 $r=33.3\%$ 时的精度，这一召回率水平上的精度即是在其之上的召回率水平所对应精度的最大值。在召回率水平 40%、50% 和 60% 上，插值后的精度是 25%，对应的召回率水平是 $r=66.6\%$，这一召回率水平上的精度即是在其之上的召回率水平所对应精度的最大值。在召回率水平 70%、80%、90% 和 100% 上，插值精度是 20%，对应于召回率水平 $r=100\%$，这一召回率水平高于所有已知精度对应的召回率。



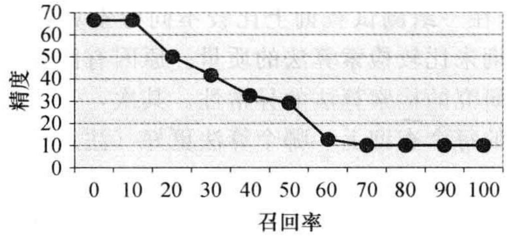
召回率	精度
0	33.3
10	33.3
20	33.3
30	33.3
40	25
50	25
60	25
70	20
80	20
90	20
100	20

137 图 4-3 对于查询 q_2 在 11 个标准召回率水平上的插值精度，分别用图和表显示

在上述的例子中，精度和召回率数值是针对单一查询计算的。然而，检索算法通常是通过运行多个不同的测试查询来进行评价的。在这种情况下，对于每个测试查询，都要生成一个不同的精度-召回率曲线。为了评价一个算法在一组 N_q 个测试查询上的检索质量，我们采用如下公式计算每个召回率水平上的平均精度。

$$\overline{P}(r_j) = \sum_{i=1}^{N_q} \frac{P_i(r_j)}{N_q} \tag{4-4}$$

其中 $\overline{P}(r_j)$ 是在召回率水平 r_j 的平均精度，$P_i(r_j)$ 是第 i 个查询在召回率水平 r_j 上的精度。举例来说，图 4-4 显示了在查询 q_1 和 q_2 上的平均精度和召回率。



召回率	精度
0	66.6
10	66.6
20	49.9
30	41.6
40	32.5
50	29.1
60	12.5
70	10
80	10
90	10
100	10

图 4-4　11 个标准召回率水平上的插值精度对查询 q_1 和 q_2 进行平均，分别用图和表显示

在一组测试查询上计算的平均精度-召回率数值通常用于比较不同算法之间的检索质量，例如可以比较新提出的检索算法和经典向量空间模型。图 4-5 显示了两个不同的检索算法的平均精度-召回率数值。在这个例子中，第一个算法在低召回率水平上有更高的精度，而第二个算法在高召回率水平上更为优越。因此，第一个算法将更适合于 Web，而另一个在法律和健康领域中应用得更好，因为通常需要较高召回率来解决法律和医疗案例。关于使用平均精度-召回率曲线来比较不同排序函数的真实例子，读者可参考附录 A。此外，我们应该注意，在某些情况下，用于比较不同排序的通用方法是使用曲线下面积（Area Under the Curve，AUC)，曲线下面积的值越大表明质量越好。

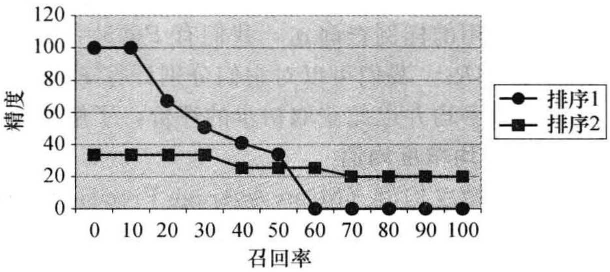


图 4-5　两个不同检索算法的平均精度-召回率数值

平均精度-召回率数值如今成为信息检索系统的标准评价指标，在信息检索文献中被广泛采用。138 它们是实际可用的，因为它们使得我们能够量化地评价（检索）结果的质量和检出相关文档所占的比例。而且，它们的表示简单直接，并且能结合在一条单一的曲线中。然而，精度-召回率数值也有缺点，有些文献对其被过多使用的情况也提出了批评。我们稍后再来讨论该问题。我们首先讨论用单一数值来概括精度-召回率数值的技术。

精度和召回率的合理性

精度和召回率已经被广泛地用来评价检索算法的质量。然而，有些更为细致的工作揭示了这两个指标所存在的问题 [931，1326，1552]。第一，对查询最大召回率的适当估计需要数据集中所有文档的详细知识。对于大规模文档集，我们无法获得这样的知识，也就意味着召回率无法准确估计。第二，精度和召回率是相关联的指标，它们描述了检出文档集不同方面的性质。在许多情况下，使用结合了召回率和精度的单一指标可能更合适。第三，精度和召回率能够度量在批处理状态下对一组查询进行处理的效果。然而，对于现代的系统来说，交互性（而不是批处理）是检索过程的关键特性。因此，对检索过程加以量化的信息性（informativeness）测度现在可能是更合适的。第四，当检出文档满足某个线性顺序时，精度和召回率是容易定义的。然而，对于只需要弱偏序关系的系统来说，精度和召回率可能是不适合的。

尽管有这些缺点，但精度和召回率仍然被广泛采用，因为在给定一个参考集的条件下，

它们易于产生且允许直接比较不同的排序策略。

4.3.2 单值总结：$P@n$，MAP，MRR，F

平均精度-召回率数值可以用于在一组测试查询上比较不同检索算法的质量。然而，在
[139] 有些情况下，我们希望针对单个查询来比较检索算法的质量。原因有两方面。首先，基于多个查询的平均精度可能掩盖了正在研究的检索算法的异常性。其次，当比较两个算法时，我们可能想要研究在给定样例查询集的每个查询上，哪个算法更好（注意单一查询的情况很容易被平均精度的计算结果所掩盖）。在这种情况下，可以采用单一（针对每个查询）的精度值。通常，这个单一的总结性数值是在一个给定召回率水平上的精度。举例来说，当我们观察到第一篇相关文档时，我们就能评测此时的精度并把这个精度作为总结性的单一数值。当然，这并不是一个很好的方法。我们下面将讨论一些更为有趣的策略。

1. 前 n 平均精度：$P@n$

在 Web 搜索引擎下，我们常常会计算获取 5 篇或者 10 篇文档时的平均精度（不管它们是相关的还是不相关的）。n 的典型值是 $P@5$(precision at 5)、$P@10$(precision at 10) 和 $P@20$(precision at 20)。它们近似地反映了用户对检索结果的印象，并且是建立在人们很少翻阅 Web 检索结果第二页的这一事实基础之上（见 7.2.1 节）。相关文档越集中于排序的顶部，用户的印象就越正面。

正如 Cleverdon 在 Cranfield-2 实验中所说的那样（见 4.2 节），大部分搜索不需要高的召回率。相反地，绝大部分的用户只需要顶部的几篇相关文档。$P@5$ 和 $P@10$ 提供了可靠的指标来评价 Web 搜索引擎的用户是否在排序的顶部得到了相关文档。举例来说，对于我们已经使用的样例查询 q_1，我们有 $P@5=40\%$ 和 $P@10=40\%$。而且，给定两个 Web 排序算法 $\mathcal{R}_1$ 和 $\mathcal{R}_2$，我们可以对它们分别计算 $P@5$ 和 $P@10$ 的数值，我们还可以在 100 个样例查询上做平均并以此获取初步的评价，了解哪个算法在用户眼中更好。

2. 平均精度均值

平均精度均值（Mean Average Precision，MAP）的主要想法是产生一个关于排序的总结性的单一数值，而这是通过对每个新观察到的相关文档计算精度并做平均获得的。

定义 设 R_i 为查询 q_i 对应的相关文档集合，并且 $|R_i|$ 表示其大小（即查询 q_i 相关文档的数量）。设 $R_i[k]$ 表示 R_i 中第 k 篇文档。那么，$P(R_i[k])$ 是在查询 q_i 的排序队列中观察到文档 $R_i[k]$ 的概率。如果这篇文档未能检出，那么 $P(R_i[k])$ 被当做 0(这在实际的搜索中经常出现，虽然它是未定义的，但是我们能假设其值足够小，并近似为 0)。

MAP_i，查询 q_i 的平均精度，定义为

[140]

$$MAP_i = \frac{1}{|R_i|} \sum_{k=1}^{|R_i|} P(R_i[k]) \tag{4-5}$$

MAP，在一组查询上的平均精度均值，定义为

$$MAP = \frac{1}{N_q} \sum_{i=1}^{N_q} MAP_i \tag{4-6}$$

其中 N_q 是查询的总数目。举例来说，以图 4-2 中的查询 q_1 作为例子。在每个新的相关文档被观察到后的精度值分别是 1、0.66、0.5、0.4 和 0.33。因此，查询 q_1 的平均精度是

$$MAP_1 = \frac{1+0.66+0.5+0.4+0.33+0+0+0+0+0}{10} = 0.28$$

注意不是所有相关文档都出现在排序的队列中，因此不在其中的文档的精度设为 0。对

于图 4-3 中的查询 q_2，在观察到每个新的相关文档后的精度分别是 0.33、0.25 和 0.20。因此，查询 q_2 的平均精度是

$$MAP_2 = \frac{0.33+0.25+0.20}{3} = 0.26$$

对于由这两个查询组成的集合，MAP 值是

$$MAP = \frac{MAP_1 + MAP_2}{2} = 0.27$$

3. *R* 精度

这里的想法是在排序的第 R 个位置计算精度，从而为该排序生成一个总结性的单一值，其中 R 是当前查询的相关文档的总数（即集合 R_q 中文档的个数）。举例来说，考察图 4-2 和图 4-3 中的样例查询 q_1 和 q_2。对于查询 q_1，相关文档的总数是 10(即 R_1 的大小)，在排序的前 10 篇文档中有 4 篇是相关文档。因此，q_1 的 R 精度值（R-Precision）是 0.4。对于查询 q_2，总的相关文档个数是 3(即 R_2 的大小)，在排序的前 3 篇文档中有一篇相关文档。这样，q_2 的 R 精度值是 0.33。

R 精度指标是一个很有用的参数，可以用来在实验中观察某个算法对于单个查询的效果。此外，人们也可以在所有的查询上计算平均 R 精度值。然而，使用单一数值来总结一个检索算法在多个查询上的效果可能是相当不准确的。

4. 精度直方图

多个查询的 R 精度值可以按如下方式来比较两个算法的检索质量。设 $RP_A(i)$ 和 $RP_B(i)$ 是检索算法$\mathcal{R}_A$ 和$\mathcal{R}_B$ 对第 i 个查询的 R 精度值。可以按如下的方式定义它们的差值：

$$RP_{A/B}(i) = RP_A(i) - RP_B(i) \tag{4-7}$$ 141

$RP_{A/B}(i)$ 的值等于 0，表明这两个算法对于第 i 个查询（就 R 精度而言）有等同的检索质量。$RP_{A/B}(i)$ 的值为正数，表明算法$\mathcal{R}_A$ 的检索质量更好，而负数则表明算法$\mathcal{R}_B$ 更好。图 4-6 显示了两个假设的检索算法在 10 个样例查询上的 $RP_{A/B}(i)$ 值。算法 A 在 8 个查询上更好，而算法 B 在另外两个查询上更好（编号 4 和 5)。这种柱状图称为**精度直方图**（precision histogram)，使我们可以通过可视化的观察，快速地比较两个算法的检索质量。

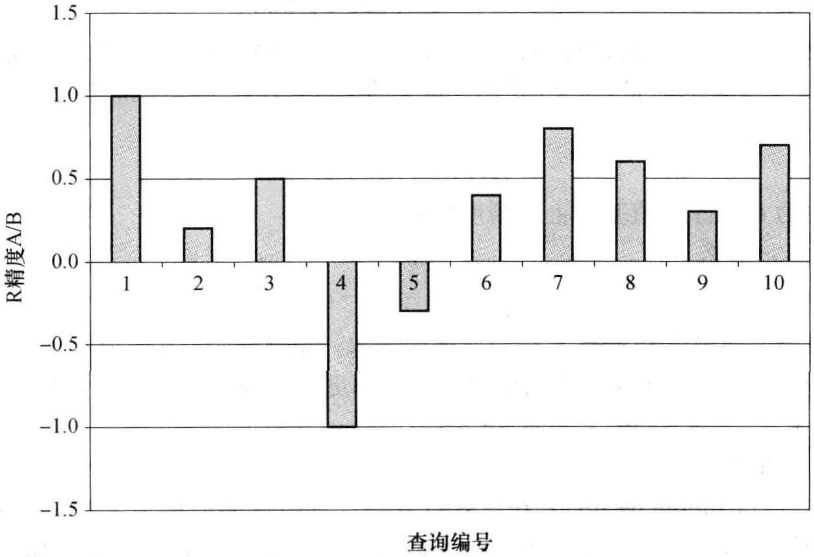


图 4-6 10 个假设查询的精度直方图

5. 平均排序倒数

有时候对于给定的查询或者任务，我们特别关注第一个正确的答案。对于问答系统（Question-Answering，QA）来说尤其是这样，因为其目标是检索出正确回答问题的短文本片段，而不是一个排序的答案集。对于 Web 查询中的 URL 查询和主页查询也是相同的情况，用户指定一个 URL 或者一个指向某个主页的引用，并特别关注第一个正确的答案。在这些情形中，我们期望指标能够倾向于那些第一个正确答案有较高排名的结果。

142 **定义** 设$\mathcal{R}_i$是相对于查询q_i的排序。设$S_{correct}(\mathcal{R}_i)$是返回在$\mathcal{R}_i$中第一个正确答案位置的函数。给定一个排序位置阈值$S_h$，$\mathcal{R}_i$的排序倒数定义为

$$\begin{cases}\frac{1}{S_{correct}(\mathcal{R}_i)} & S_{correct}(\mathcal{R}_i) \leqslant S_h \\ 0 & S_{correct}(\mathcal{R}_i) > S_h\end{cases}$$

也就是说，如果第一个正确答案出现在排序中 S_h 之后的位置，那么排序倒数是零。对于由 N_q 个查询组成的集合 Q 来说，平均排序倒数（Mean Reciproach Rank，MRR）是所有排序倒数的均值，即

$$MRR(Q) = \frac{1}{N_q} \cdot \sum_{i=1}^{N_q} \frac{1}{S_{correct}(\mathcal{R}_i)} \tag{4-8}$$

MRR 是倾向于那些第一个正确的结果出现在排序顶部的指标。它总是介于 0～1 之间，并且和平均精度有紧密的相关性，这是好的特性。同时 MRR 也表现出一些缺点。比如它只考察第一个正确的结果，且只能取一些离散值，例如，1、1/2、1/3 是分别对应于排序位置 1、2、3 的值。尽管如此，MRR 是一个有用的指标，用来评测那些非常注重第一个正确答案的情况，例如 QA 会话、URL 和主页查询。

6. E 值

E 值（E-Measure）最初是由 van Rijsbergen[1624] 提出的。其想法是结合精度和召回率，允许用户指定它们是否对召回率或者精度更为关注。E 值按如下方式定义

$$E(j) = 1 - \frac{1+b^2}{\frac{b^2}{r(j)} + \frac{1}{P(j)}} \tag{4-9}$$

其中 $r(j)$ 是在排序中第 j 个位置的召回率，$P(j)$ 是在排序中第 j 个位置的精度，$0 \leqslant E(j) \leqslant 1$ 是排序中第 j 个位置的 E 值，$b \geqslant 0$ 是用户定义的参数，反映了精度和召回率的相对重要性。

如果 $b=0$，那么

$$E(j) = 1 - P(j)$$

这表明低数值的 b 使得 E 值基本上是精度的函数。

如果 $b \to \infty$，那么

$$\lim_{b \to \infty} E(j) = 1 - r(j)$$

这表明高数值的 b 使得 E 值基本上是召回率的函数。也就是说，b 的值小于 1 表明用户对于精度更感兴趣，而 b 的值大于 1 表明用户对召回率更感兴趣。对于 $b=1$，E 值变成了 F 值或

143 者调和平均。

7. F 值：调和平均

调和平均（Harmonic Mean），在信息检索的背景中更多地称为 F 值（F-Measure），它提供了另一种把精度和召回率结合为一个数值的方法，大家可能对此感兴趣 [1455]。它是按照如下的方式计算的：

$$F(j)=\frac{2}{\frac{1}{r(j)}+\frac{1}{P(j)}} \tag{4-10}$$

其中 $r(j)$ 是在排序列表第 j 个位置的召回率，$P(j)$ 是在排序中第 j 个位置的精度，$F(j)$ 是排序中第 j 个位置的调和平均。注意到，

$$F(j)=1-E(j)$$

如果在计算 $E(j)$ 中，取 $b=1$。也就是说，函数 F 是 E 值的补值。当没有相关文档检出时，其值为 0；当所有排序文档都是相关文档时，其值为 1。而且，仅当召回率和精度都较高时，它才有较高的数值。也就是说，确定 F 最大值的过程可以解释为在精度和召回率之间找到最为合适的平衡。

调和平均指标也常常用来评价文本分类算法。在用于这一目的时，它称为 F_1 值，在第 8 章中讨论。

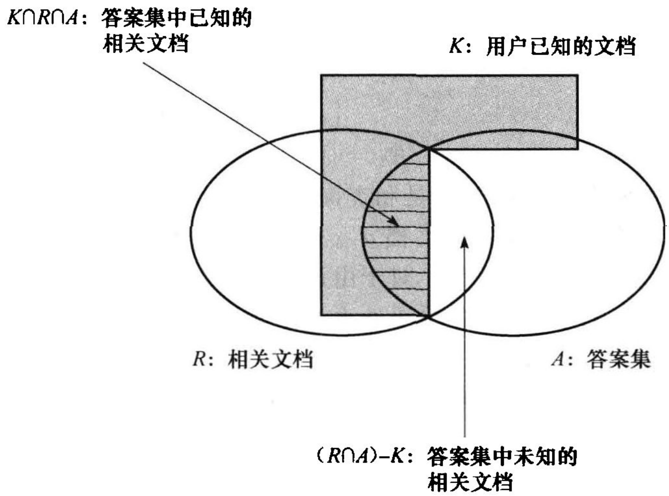
8. 摘要统计表

单一的数值也可以存储在表中用于提供统计性的总结。这些总结性统计表可以包括各种内容：在任务中使用的查询的个数，所有查询检出的文档的总数，所有查询检出的相关文档的总数，所有查询由专家评判的相关文档的总数。

4.3.3 面向用户的指标

精度和召回率是基于以下的假设：查询的相关文档集合是不变的，与用户无关。然而，不同的用户可能对哪些文档是相关的，哪些文档是不相关的，有着不同的解释。为了处理这个问题，已经提出了诸如覆盖率（coverage ratio）、新颖率（novelty ratio）、相对召回率（relative recall）、召回代价（recall effort）[931] 等面向用户的指标值。

如前所述，假设有一个参考集，一个样例信息需求 I 和一个待评测的检索算法。对于 I，设 R 是相关文档的集合，A 是检出答案的集合。同时，设 K 为数据集中用户已知的文档集合，$|K|$ 为其大小。集合 $K\cap R\cap A$ 是集合 K、R 和 A 的交集，是由用户已知的、相关的和检出的文档组成的。而且，集合 $(R\cap A)-K$ 是由用户未知的、检出的相关文档组成的。图 4-7 说明了这一情况。覆盖率定义为已知且相关的文档在答案集中所占的比例，也就是



144

图 4-7 对于给定的样例信息需求的覆盖率和新颖率

$$coverage=\frac{|K\cap R\cap A|}{|K\cap R|} \tag{4-11}$$

新颖率定义为用户未知的相关文档在答案集中所占的比例，也就是

$$novelty=\frac{|(R\cap A)-K|}{|R\cap K|} \tag{4-12}$$

高覆盖率表明系统找到了用户期望看到的大部分相关文档。高新颖率表明系统（向用户）展

示了许多之前未知的新的相关文档。

另外，我们还可以定义了两个指标：相对召回率和召回代价。*相对召回率*是（由系统）找到的相关文档数量与用户希望找到的相关文档数量的比率。如果用户找到了与他期望一样多的文档，他就停止搜索，则相对召回率等于1。*召回代价*是用户希望找到的文档数量与搜寻相关文档的过程中检查到的文档数量的比率。

4.3.4 折扣累积增益

精度和召回率尽管被广为使用，但是它们仅允许二元的相关性评价，因此可能被野值（即在排序靠后位置找到的相关文档）严重影响。因此，它们可能会无法区分能在排序的顶部检出高度相关文档的模型与仅能检索出轻度相关文档的模型之间的区别。这样的局限性可
145 以通过采用分级相关性评价以及能够有效结合这些评价的指标来克服。这些指标包括现在我们要讨论的折扣累积增益（Discounted Cumulated Gain，DCG）。我们的讨论基于 Järvelin 和 Kekäläinen[828，829] 的工作。注意有些作者使用 Cumulative 而不是 Cumulated，但是我们还是决定采用原来的名称。

1. 基本折扣累积增益

当检查查询的结果时，我们可以观察到两个重要的现象：

1）在排序的顶部我们更希望是高度相关的文档，而不是轻度相关文档；

2）出现在排序底部的相关文档的价值不高。

对于第一个现象，我们通过采用分级相关评价的方式，并且利用排序中的相关文档计算*累积增益*（cumulated gain）来提高评价质量。让我们用一个例子来说明。

假设一组测试查询集合的检索结果是由专家评审的，并且被分为0～3级，3表示强相关性，而0表示文档是不相关的。举例来说，对于在4.3.1节中用做样例的查询 q_1 和 q_2，假设分级相关度分数是如下所示的结果：

$$R_1 = \{[d_3,3],[d_5,3],[d_9,3],[d_{25},2],[d_{39},2],$$
$$[d_{44},2],[d_{56},1],[d_{71},1],[d_{89},1],[d_{123},1]\}$$
$$R_2 = \{[d_3,3],[d_{56},2],[d_{129},1]\}$$

也就是说，文档 d_3 是和查询 q_1 高度相关的，文档 d_{56} 只是轻度相关的。而文档 d_3 对于查询 q_2 是高度相关的，文档 d_{129} 只是轻度相关的。给定这些相关性评价结果，排序算法的结果可以按如下方式评价。对于由该算法为查询 q 生成的前10～20个结果中的每一个，我们根据由专家所做的评估给予一个分级的相关性分数。这个相关性分数的排序列表称为*增益向量*（gain vector）G。举例来说，假设我们考察由查询 q_1 和 q_2 生成排序的前15篇文档，如4.3.1节所示。给定如上的分级相关性评价，对于这些查询的增益向量 G_1 和 G_2 是

$$G_1 = (1,0,1,0,0,3,0,0,0,2,0,0,0,0,3)$$
$$G_2 = (0,0,2,0,0,0,0,1,0,0,0,0,0,0,3)$$

通过把排序中任何一点的分级相关性分数（即增益）累加，我们即获得了由这个算法产生的*累积增益*（Cumulated Gain，CG）指标。举例来说，对于查询 q_1，第一个位置的累积增益值是1，在第二个位置是1+0，在第三个位置是1+0+1，以此类推。因此，q_1 和 q_2 的*累积增益向量*为

$$CG_1 = (1,1,2,2,2,5,5,5,5,7,7,7,7,7,10)$$
$$CG_2 = (0,0,2,2,2,2,2,3,3,3,3,3,3,3,6)$$

146 举例来说，CG_1 在位置8的累积增益值是5。

定义 给定测试查询 q_j 的增益向量 G_j，它对应的累积增益函数 CG_j 被定义为

$$CG_j[i]=\begin{cases}G_j[1] & i=1\\ G_j[i]+CG_j[i-1] & i>1\end{cases} \tag{4-13}$$

其中 CG_j $[i]$ 表示查询 q_j 的排序中第 i 个位置的累积增益。

对于在本节之前提到的第二个现象，即在排序底部的相关文档的价值不高，我们引入一个折扣因子以减少它对增益的影响。一个简单的增益因子是排序位置的对数函数（log）。如果我们考察以 2 为底的对数，那么这个折扣因数在位置 2 是 $\log_2 2$，在位置 3 是 $\log_2 3$，在位置 4 是 $\log_2 4$，以此类推。通过在第 i 位对于增益 $G_j[i]$ 除以对应的折扣因子，可以得到折扣累积增益。

定义 给定测试查询 q_j 的增益向量 G_j，其对应的折扣累积增益函数 DCG_j 可定义为

$$DCG_j[i]=\begin{cases}G_j[1] & i=1\\ \dfrac{G_j[i]}{\log_2 i}+DCG_j[i-1] & i>1\end{cases} \tag{4-14}$$

其中 $DCG_j[i]$ 表示查询 q_j 的排序在第 i 个位置的折扣累积增益。

对于样例查询 q_1 和 q_2，DCG 向量（舍入值）如下

$DCG_1=(1.0,1.0,1.6,1.6,1.6,2.8,2.8,2.8,2.8,3.4,3.4,3.4,3.4,3.4,4.2)$

$DCG_2=(0.0,0.0,1.3,1.3,1.3,1.3,1.3,1.6,1.6,1.6,1.6,1.6,1.6,1.6,2.4)$

我们注意到折扣累积增益值不太受排序底部的相关文档的影响，这正是引入折扣因子所期望达到的效果。在上面的例子中，在第 15 位出现的一个高度相关的文档使得 DCG_1 相对于 CG_1 的升幅要小得多。进一步注意到通过在对数函数采用更大的底数，可以加强折扣因子的效果。但是为了简单起见，我们把讨论限制在以 2 为底的对数函数中。

2. DCG 曲线

为了在一组测试查询上产生 CG 和 DCG 曲线，我们需要在所有查询上加以平均，具体如下所示。

定义 给定一个由 N_q 个测试查询组成的集合，在这组测试查询上的平均 $\overline{CG}$ $[i]$ 和 $\overline{DCG}$ $[i]$ 因子可以按如下方式计算。

$$\overline{CG}[i]=\frac{1}{N_q}\sum_{j=1}^{N_q}CG_j[i]$$
$$\overline{DCG}[i]=\frac{1}{N_q}\sum_{j=1}^{N_q}DCG_j[i] \tag{4-15}$$

147

举例来说，对于样例查询 q_1 和 q_2，这些平均结果（舍入值）为

$$\overline{CG}=(0.5,0.5,2.0,2.0,2.0,3.5,3.5,4.0,4.0,5.0,5.0,5.0,5.0,5.0,8.0) \tag{4-16}$$

$$\overline{DCG}=(0.5,0.5,1.5,1.5,1.5,2.1,2.1,2.2,2.2,2.5,2.5,2.5,2.5,2.5,3.3) \tag{4-17}$$

在已经计算了平均 CG 和 DCG 因子之后，就可以通过把排序位置从 1 变动到一个预设的阈值来绘制平均曲线。在上面的例子中，这个阈值设为 15，在 Web 中其值通常设为 10。

图 4-8 显示了由式（4-16）和式（4-17）计算的 $\overline{CG}$ 和 $\overline{DCG}$ 向量对应的 CG 和 DCG 曲线。我们注意到 CG 和 DCG 数值在开始的时候迅速增长，然后变得平缓。而且，由于折扣因子的影响，DCG 比 CG 增长得慢。在位置 15，我们观察到两个数值的突然增长，但这是由于在查询 q_1 和 q_2 的排序的底部都出现了一个高度相关的文档。当我们在大量的查询上对 CG

和 DCG 分数进行平均时，这个现象应当就会消失。而且，当这些平均值是在大规模的查询上计算时，其曲线会更接近抛物线的形状（在水平轴附近），开始增长迅速，而后变得平缓。

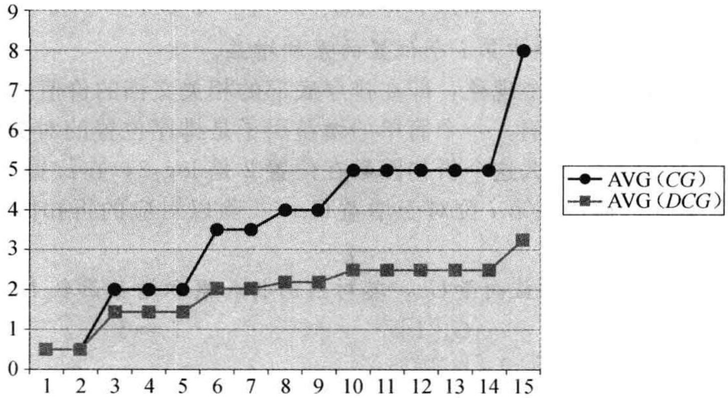


图 4-8 对应于$\overline{CG}$和$\overline{DCG}$向量的 CG 和 DCG 曲线，通过式（4-16）和式（4-17）计算

3. 理想的 CG 和 DCG 指标

精度和召回率数值是相对于相关文档集计算的，它们能直接用于比较不同的算法。而前面定义的 CG 和 DCG 数值不是相对于任何基准计算的，这意味着人们可能会误以为能够用它们来直接比较不同的检索算法。解决这个问题的一种方法是计算归一化的 CG 和 DCG 指标，这需要定
148 义用于归一化的基准。这个基准就是我们接下来要讨论的理想的 CG 和 DCG 指标。

定义 对于给定的测试查询 q，假设由专家生成的相关文档集包含 n_3 篇相关度为 3 分的文档，n_2 篇相关度为 2 分的文档，n_1 篇相关度为 1 分的文档，n_0 篇相关度 0 分的文档（即判断为不相关）。理想的增益向量 IG 是通过对所有相关分数按照降序排列生成的，如下所示：

$$IG = (3,\cdots,3,2,\cdots,2,1,\cdots,1,0,\cdots,0)$$

也就是说，IG 向量是由 n_3 个值为 3，n_2 个值为 2，n_1 个值为 1，以及 n_0 个值为 0 的数值组成。

举例来说，对于样例查询 q_1 和 q_2 而言，我们有

$$IG_1 = (3,3,3,2,2,2,1,1,1,1,0,0,0,0,0)$$
$$IG_2 = (3,2,1,0,0,0,0,0,0,0,0,0,0,0,0)$$

给定 IG 向量，理想的 $CG(ICG)$ 和理想的 $DCG(IDCG)$ 向量可以类似于 CG 和 DCG 进行计算。举例来说，对于样例查询 q_1 和 q_2，理想 CG 向量是如下的形式

$$ICG_1 = (3,6,9,11,13,15,16,17,18,19,19,19,19,19,19)$$
$$ICG_2 = (3,5,6,6,6,6,6,6,6,6,6,6,6,6,6)$$

理想 DCG 向量是如下的形式

$$IDCG_1 = (3.0,6.0,7.9,8.9,9.8,10.5,10.9,11.2,11.5,11.8,11.8,11.8,11.8,11.8,11.8)$$
$$IDCG_2 = (3.0,5.0,5.6,5.6,5.6,5.6,5.6,5.6,5.6,5.6,5.6,5.6,5.6,5.6,5.6)$$

进一步地，平均$\overline{ICG}$值和平均$\overline{IDCG}$值可以按如下方式计算。

$$\overline{ICG}[i] = \frac{1}{N_q}\sum_{j=1}^{N_q} ICG_j[i]$$
$$\overline{IDCG}[i] = \frac{1}{N_q}\sum_{j=1}^{N_q} IDCG_j[i] \qquad (4\text{-}18)$$

举例来说，对于样例查询 q_1 和 q_2，$\overline{ICG}$和$\overline{IDCG}$向量是如下的形式

$\overline{ICG}$ = (3.0,5.5,7.5,8.5,9.5,10.5,11.0,11.5,12.0,12.5,12.5,12.5,12.5,12.5,12.5)

$\overline{IDCG}$ = (3.0,5.5,6.8,7.3,7.7,8.1,8.3,8.4,8.6,8.7,8.7,8.7,8.7,8.7,8.7)

注意到理想曲线确立了可达的最高检索质量，对于一个算法，通过将其 CG 和 DCG 的平均曲线和理想平均曲线进行比较，就能更深入地了解还有多少提升的空间。

4. 归一化折扣累积增益

精度和召回率数值可以直接与各召回率上均是 100%精度的理想曲线进行比较。然而，DCG 数值不是相对于任何理想曲线建立的（即使我们能绘制出理想的 DCG 曲线来理解可达的最大检索质量）。其结果是我们难以对两个不同的排序算法直接比较其 DCG 曲线。而这可以通过归一化 DCG 指标来修正。 149

定义 给定一组 N_q 个测试查询的平均 CG、ICG、DCG 和 $IDCG$ 曲线，归一化 CG 和 DCG 指标是如下的形式

$$NCG[i] = \frac{\overline{CG}[i]}{\overline{ICG}[i]}$$

$$NDCG[i] = \frac{\overline{DCG}[i]}{\overline{IDCG}[i]} \quad (4\text{-}19)$$

举例来说，对于样例查询 q_1 和 q_2，NCG 和 $NDCG$ 向量是如下的形式

NCG =(0.17,0.09,0.27,0.24,0.21,0.33,0.32,
0.35,0.33,0.40,0.40,0.40,0.40,0.40,0.64)

$NDCG$ =(0.17,0.09,0.21,0.20,0.19,0.25,0.25,
0.26,0.26,0.29,0.29,0.29,0.29,0.29,0.38)

归一化数值也可以对单一的查询进行计算，但这里我们关注整个测试集的归一化数值。在 NCG 和 $NDCG$ 曲线下面积代表了排序算法的质量。这个面积越大，就认为结果越好。这样，归一化数值就可以用来比较两个不同的排序算法。而且它们也能用来比较在排序中某个给定位置上的检索质量。举例来说，给定两个排序算法 $\mathcal{R}_A$ 和 $\mathcal{R}_B$，我们能在位置 10 比较它们的归一化指标，即我们直接比较 $NDCG_A[10]$ 和 $NDCG_B[10]$。这类似于为两个算法在排序的位置 10 比较精度数值，即 $P@10$。

5. 对于 DCG 指标的讨论

CG 和 DCG 指标旨在考虑多层次的相关性评价，而不是更常见的用于精度和召回率指标的二元相关性评价。它的优点是能够区别高度相关的文档和轻度相关的文档，因为前者有着更高的相关性分数。其内在的缺点是要生成多层相关性水平更难且更耗时，并且由于多层相关性分数往往依赖于主观性解释，使其更容易产生错误。

尽管有这些内在的困难，但 CG 和 DCG 指标还是表现出了许多优势：1）它们能够系统地结合文档排序和相关性分数；2）累积增益提供了在排序中任意位置上的单值检索质量指标，且独立于召回率；3）累积增益强调了相关文档在排序中某个特定位置所产生的增益，这使得该指标对野值有更好的免疫能力；4）折扣累积增益能够减少在排序底部发现的相关文档的影响。因此，在需要高度准确地评价那些复杂、成熟，且有着相近检索质量的算法时，除了已经很完善的精度和召回率指标外，CG 和 DCG 指标也是非常有竞争力的一种选择。

4.3.5 二元偏好

Cranfield 评价范式是基于对相关文档完全了解的情况下建立的，即测试集中所有的文档是相对于每个测试查询进行评价的，且每个文档-查询对被赋予了一个二元的相关性评价。这在小文档集上可以完成得很好，但是对诸如 TREC(见 4.4.1 节）等大文档集是不实际的。 150

对于类似于 TREC 那样的大文档集，通常是使用聚合（pooling）方法来代替原有的方案。它是通过整合不同的检索算法返回的排名靠前的结果，将其存在库中，并为它们生成相关性评价。对于小于 200 万篇文档的数据集，库的典型规模介于 1000～2000 之间。在这样的条件下，已经证实基于这个结果库的相关性评价是可靠的，并可以用来有效地比较不同系统的检索质量。也就是说，Cranfield 范式在 TREC 文档集的情况下仍是有效的，该文档集的规模要比需要赋予相关性评价的结果库的规模大三个数量级。

我们在 Web 中会遇到不同的情况，Web 数据集是由数以十亿计的文档组成的。对于这种规模的文档集，无法保证聚合方法能够可靠地比较不同的 Web 检索算法。其中潜在的主要问题是，若对 Web 文档集使用聚合方法，则会有许多没有见到的文档视为不相关。对于小文档集，如果相关性评价的规模有限，也会产生类似的问题。因此，除了精度和召回率之外，我们还需要设计不同的指标，专门对含有不完整信息的结果进行评价。这就是提出二元偏好（Binary Preferences，BPREF）指标的目的。我们下面的讨论是基于 Buckley 和 Voorhees[292] 的工作。

1. *Bpref*

诸如精度-召回率以及 $P@10$ 等指标对于明确评价为不相关的文档和没有被检出的文档而言是没有区别的，因为它们都被认为是不相关的。对于规模非常大的文档集来说，这会成为一个不小的问题，因为对于单一的查询，有太多的文档没有检索出来。一个巧妙应对这个问题的方法是采用偏序关系，即利用任意两篇检出文档之间的偏序关系来定义这个指标，而不是直接使用排名位置。这就是产生了二元偏好（$Bpref$）指标的基本想法。

$Bpref$ 度量了对于某个特定的信息需求，由专家确定的出现在相关文档之前的不相关文档的数量。这个测度被称为 $Bpref$，因为这种倾向关系是二元的，即该评价就是对于给定的信息需求，判断文档 d_j 是否比文档 d_k 更好。例如，对于某个信息需求，任何相关文档要比不相关文档好。

图 4-9 显示了在 $Bpref$ 的计算中所要考虑的文档集合。集合 J 是由专家对某个信息需求做出评价的所有文档组成的。这个集合包含了我们认为相关的文档集合 R，大小为 $|R|$，以及我们认为不相关的文档集合 $J-R$，大小是 $|J-R|$。集合 $R\cap A$ 是由答案集中的相关文档组成的，而集合 $(J-R)\cap A$ 是由答案集中的不相关文档组成的。

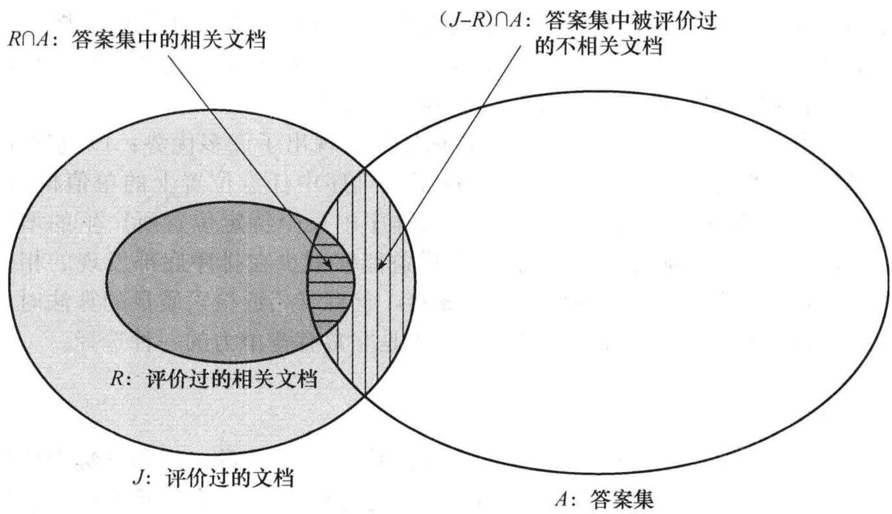


图 4-9 用于计算 $Bpref$ 的集合

一种直接定义 $Bpref$ 指标的可能方法是计算在排序中出现在相关文档前且属于集合 $J-R$ 的文档的数量。然而，这会使得该指标依赖于查询所对应的集合的大小，从而使我们无法有效地计算多个查询的平均值。这个问题是按照如下的方法解决的。 151

定义　给定信息需求 I，定义如图 4-9 所示的集合 J、R 和 A。如果 $\mathcal{R}_A$ 是 A 中答案的排序，那么令 $s_{A,j}$ 表示文档 d_j 在 $\mathcal{R}_A$ 的位置。设 $[(J-R)\cap A]_{|R|}$ 是由 $\mathcal{R}_A$ 中前 $|R|$ 篇被评为不相关的文档所组成的集合。定义函数 $C(\mathcal{R}_A, d_j)$ 是出现在 $\mathcal{R}_A$ 的前 $|R|$ 篇不相关文档中，且排在文档 d_j 之前的不相关文档的个数。也就是

$$C(\mathcal{R}_A, d_j) = \|\{d_k \mid d_k \in [(J-R)\cap A]_{|R|} \wedge s_{A,k} < s_{A,j}\}\|$$

排序 $\mathcal{R}_A$ 的 $Bpref$ 定义为

$$Bpref(\mathcal{R}_A) = \frac{1}{|R|}\sum_{d_j\in(R\cap A)}\left(1-\frac{C(\mathcal{R}_A, d_j)}{\min(|R|, |(J-R)\cap A|)}\right) \tag{4-20}$$

注意集合 $(J-R)\cap A$ 可能少于 $|R|$ 篇文档。在这种情况下，集合 $[(J-R)\cap A]_{|R|}$ 等同于 $(J-R)\cap A$，并有少于 $|R|$ 篇的文档。上面的 $Bpref$ 的公式是 [292] 中原公式的变体，在 [1500] 中提出。作者修正了原公式中的错误，现在这个公式用于 TREC 实验当中 [1500]。

对排序中的每篇相关文档 d_j，$Bpref$ 累加了一个权重，该权重随着在排序中位于该文档之前的不相关文档数量的增加而减小。举例来说，如果所有的 $(J-R)\cap A$ 的文档在排序中都排在 d_j 之前，累加的权重是 0。也就是说，当观察到 $(J-R)\cap A$ 中所有的文档时，在排序中后面看到的相关文档就不影响指标了。如果 $(J-R)\cap A$ 中不存在排在 d_j 之前的文档，则累加的权重是 1。在累加了所有权重后，其和会被归一化。 152

因为相关文档对应的权重会被归一化，并假设所考虑的不相关文档的数目和相关文档的最大数目相同，二元偏好指标在不完整信息的情况下仍能保持稳定，可用于在非常巨大的文档集上比较不同的检索算法。

2. *Bpref*-10

由于 $Bpref$ 倾向于用在信息不完整的条件下，而实际的情况可能是已知的相关文档的数量较小，甚至小到只有 1～2 篇。在这种情况下，指标可能会变得不稳定，尤其当用来定义 $C(\mathcal{R}_A, d_j)$ 的偏序关系数量非常小的时候。$Bpref$-10 是二元偏好的变体，旨在通过确保至少可获得 10 组偏序关系来修正这个问题。

定义　假设使用和 $Bpref$ 同样的集合表示。设 $[(J-R)\cap A]_{|R|+10}$ 是由 $(J-R)\cap A$ 中前 $|R|+10$ 篇文档组成。而且，设 $C_{10}(\mathcal{R}_A, d_j)$ 是由如下函数定义

$$C_{10}(\mathcal{R}_A, d_j)^{*} = \|\{d_k \mid d_k \in [(J-R)\cap A]_{|R|+10} \wedge s_{A,k} < s_{A,j}\}\|$$

那么，

$$Bpref_{10}(\mathcal{R}_A) = \frac{1}{R}\sum_{d_j\in(R\cap A)}\left(1-\frac{C_{10}(\mathcal{R}_A, d_j)}{\min(|R|+10, |(J-R)\cap A|)}\right) \tag{4-21}$$

4.3.6　排序相关性测度

精度和召回率使我们能够比较两个排序函数产生的结果的相关性。然而，在某些情况下 1）我们不能直接度量相关性（比方说，当我们没有参考集或者没有评测人员时）；2）我们更关注于了解一个排序函数和另一个我们熟知的函数（例如向量模型）之间有多大区别。

在这种情况下，我们关注于在缺少答案等相关信息的条件下比较由这两个排序算法产生的答案的相对顺序。这可以通过使用称为排序相关性测度（rank correlation metric）的统计函数来达到，它对每个排序函数生成的排列顺序进行比较。排序相关性测度在比较了两个排

序$\mathcal{R}_1$ 和$\mathcal{R}_2$ 之后，会产生具有下述属性的相关系数 $C(\mathcal{R}_1, \mathcal{R}_2)$：

- $-1 \leqslant C(\mathcal{R}_1, \mathcal{R}_2) \leqslant 1$。
- 如果 $C(\mathcal{R}_1, \mathcal{R}_2)=1$，那么这两个排序之间是完全一致的，即它们是相同的。
- 如果 $C(\mathcal{R}_1, \mathcal{R}_2)=-1$，那么这两个排序之间是完全不一致的，即它们是彼此反序的。
- 如果 $C(\mathcal{R}_1, \mathcal{R}_2)=0$，那么这两个排序是完全独立的。

153 • $C(\mathcal{R}_1, \mathcal{R}_2)$值的增长表示了两个排序之间的一致性增强。

这里，我们主要考虑两个排序相关性测度：斯皮尔曼系数［1511］和肯德尔系数［898］。我们的讨论基于参考文献［3，2，1052，1208］。

1. 斯皮尔曼系数

斯皮尔曼系数（Spearman Coefficient）可能是使用最多的排序相关性测度。它是基于相同文档在两个待比较的排序$\mathcal{R}_1$ 和$\mathcal{R}_2$ 中的位置区别来构建的。设 $s_{1,j}$是文档d_j 在排序$\mathcal{R}_1$ 中的位置，$s_{2,j}$是文档d_j 在排序$\mathcal{R}_2$ 中的位置。作为示例，表 4-1 展示了 10 个样例文档，它们在两个排序$\mathcal{R}_1$ 和$\mathcal{R}_2$ 中的位置，相同文档在不同排序中的位置差值，以及这些差值的平方。

如果我们观察表 4-1 中$\mathcal{R}_1$ 和$\mathcal{R}_2$ 的排序位置的差值，我们会发现它们之间的差异是很小的，这表明了这两个排序之间有较好的相关性。事实上，图 4-10 在两维的坐标平面内展示了两个排序的排序位置，并再一次说明了它们之间有紧密的联系。为了对这个相关性产生一个量化的评价，我们对每对排序差值的平方求和。在表 4-1 中的例子里，这个和是 24。一般来说，如果有 K 篇排序文档，那么排序差值平方和的最大值为

$$\frac{K \times (K^2-1)}{3}$$

这样，对于 $K=10$，如果两个排序是完全不一致的（互为反序），那么排序差值平方和的最大值是 $(10\times(10^2-1))/3$，或 330。相反，如果排序完全一致，其和是 0。

154

表 4-1 在排序$\mathcal{R}_1$ 和$\mathcal{R}_2$ 中对应的文档位置 $s_{1,j}$ 和 $s_{2,j}$，以及它们的相对差值

文档	$s_{1,j}$	$s_{2,j}$	$s_{1,j}-s_{2,j}$	$(s_{1,j}-s_{2,j})^2$
d_{123}	1	2	−1	1
d_{84}	2	3	−1	1
d_{56}	3	1	+2	4
d_6	4	5	−1	1
d_8	5	4	+1	1
d_9	6	7	−1	1
d_{511}	7	8	−1	1
d_{129}	8	10	−2	4
d_{187}	9	6	+3	9
d_{25}	10	9	+1	1
距离平方和				24

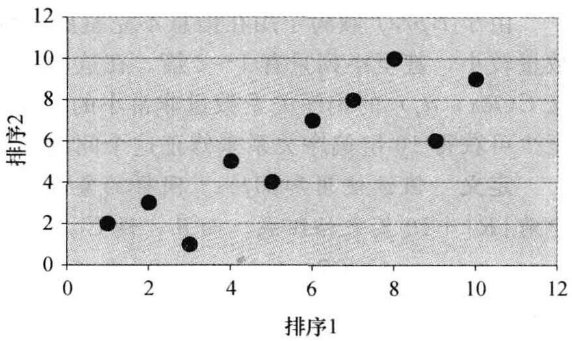


图 4-10 表 4-1 中排序$\mathcal{R}_1$ 和$\mathcal{R}_2$ 的排序位置，绘制在二维坐标系统内

现在让我们考察分式

$$\frac{\sum_{j=1}^{K}(s_{1,j}-s_{2,j})^2}{\frac{K \times (K^2-1)}{3}}$$

当两个排序完全一致的时候，其值为 0；当它们完全不一致（互为反序）的时候，值为+1。如果我们对这个分式乘以 2，值将移动到［0，+2］之间。如果我们现在把结果减去 1，其

值平移到［－1，＋1］的区间内。这个推理过程说明可以按照如下方式定义两个排序间的相关系数。

定义 设 $s_{1,j}$ 和 $s_{2,j}$ 是文档 d_j 在两个排序 $\mathcal{R}_1$ 和 $\mathcal{R}_2$ 中的排序位置。定义

$$S(\mathcal{R}_1,\mathcal{R}_2)=1-\frac{6\times\sum_{j=1}^{K}(s_{1,j}-s_{2,j})^2}{K\times(K^2-1)} \tag{4-22}$$

其中 $S(\mathcal{R}_1,\mathcal{R}_2)$ 是斯皮尔曼排序相关系数，K 表示了排序集合的大小。

对于表 4-1 中的排序，我们有

$$S(\mathcal{R}_1,\mathcal{R}_2)=1-\frac{6\times 24}{10\times(10^2-1)}=1-\frac{144}{990}=0.854$$

这表明了它们之间有很强的相关性。

一个常见的情况是在排序 $\mathcal{R}_1$ 和 $\mathcal{R}_2$ 中的文档个数是不同的。为了解决这个问题，我们需要增广每个排序，使得它们有相同的大小。这是按照以下方法实现的。

1）计算文档集合 S_{1+2}，它是由所有属于某个排序（$\mathcal{R}_1$ 或 $\mathcal{R}_2$）的文档所组成的，它的大小即是增广排序的大小，即 $K=|S_{1+2}|$。

2）用所有在 S_{1+2} 中，但不在 $\mathcal{R}_1$ 中的文档来增广排序 $\mathcal{R}_1$。这些文档被加在 $\mathcal{R}_1$ 的底部，但使用的是在 $\mathcal{R}_2$ 中的顺序。 155

3）用所有在 S_{1+2} 中，但不在 $\mathcal{R}_2$ 中的文档来增广排序 $\mathcal{R}_2$。这些文档被加在 $\mathcal{R}_2$ 的底部，但使用的是在 $\mathcal{R}_1$ 中的顺序。

下面的例子说明了这样的过程。

$$\mathcal{R}_1=(d_{123},d_{84},d_{56},d_6,d_8,d_9,d_{511},d_{129},d_{187},d_{25})$$
$$\mathcal{R}_2=(d_{56},d_{123},d_{84},d_8,d_6,d_{38},d_{48},d_{250},d_{113},d_3)$$
$$\mathcal{R}+_1=(d_{123},d_{84},d_{56},d_6,d_8,d_9,d_{511},d_{129},d_{187},d_{25},d_{38},d_{48},d_{250},d_{113},d_3)$$
$$\mathcal{R}+_2=(d_{56},d_{123},d_{84},d_8,d_6,d_{38},d_{48},d_{250},d_{113},d_3,d_9,d_{511},d_{129},d_{187},d_{25})$$

其中 $\mathcal{R}+_1$ 和 $\mathcal{R}+_2$ 分别表示增广的 $\mathcal{R}_1$ 和 $\mathcal{R}_2$ 排序。对于这些增广排序，我们有 $|S_{1+2}|=15$。这样，

$$K=15$$
$$\sum_{j=1}^{15}(s_{1,j}-s_{2,j})^2=258$$
$$S(\mathcal{R}_1,\mathcal{R}_2)=1-\frac{6\times 258}{15(15^2-1)}=0.539$$

也就是说，对于增广排序的斯皮尔曼系数等于 0.539。这表明它们之间有中等程度的正相关性。

这里作为例子的斯皮尔曼系数的计算假设了排序中任何一对排序位置总是不同的。也就是说，

$$\forall_{i,j}\,|\,i\neq j,\quad s_{1,i}\neq s_{1,j}$$

可能会产生违反这个条件的情况，例如 $s_{1,123}=1$ 和 $s_{1,84}=1$。在这个例子中，排序 $\mathcal{R}_1$ 假定文档 d_{123} 和 d_{84} 两者有相同的排序位置，即两者都在排序的顶部。在这种特殊的情况下，可以使用标准皮尔森（Pearson）相关系数来作为替代，详情见参考文献［1167］。

2. 肯德尔系数

尽管斯皮尔曼系数是一种流行且被广泛采用的相关系数指标，但是我们很难给它赋一个可操作的解释。一种替代方法是使用一种具有自然直观解释的相关系数，同时它也有着更为

简单的代数结构——肯德尔等级相关系数（Kendall Tau Coefficient）。

当我们考虑排序的相关性时，我们首先想到两个排序是否以相似的模式进行变化，即它们是否有相同的变化趋势。举例来说，给定两篇文档 d_j 和 d_k，以及它们在排序$\mathcal{R}_1$ 和$\mathcal{R}_2$ 中的位置。则这两篇文档在每个排序中的排序位置的差异是

156

$$s_{1,k} - s_{1,j}$$
$$s_{2,k} - s_{2,j}$$

如果这些差异有相同的正负号，那么我们说文档二元组 $[d_k, d_j]$ 在这两个排序中是协调的。如果它们的正负号不同，那么我们说这个文档二元组在这两个排序中是不协调的。一种简单的计算这两个排序之间的相关性强度的方法是对其中协调和不协调文档二元组的个数进行计数，并计算两者之间的差。

举例来说，对于表 4-1 中排序$\mathcal{R}_1$ 的前 5 篇文档（即假设 $K=5$），排序文档二元组是

$$[d_{123},d_{84}],\quad [d_{123},d_{56}],\quad [d_{123},d_6],\quad [d_{123},d_8],$$
$$[d_{84},d_{56}],\quad [d_{84},d_6],\quad [d_{84},d_8],$$
$$[d_{56},d_6],\quad [d_{56},d_8],$$
$$[d_6,d_8]$$

合计有$\frac{1}{2}\times5\times4$，即 10 个偏序对。也就是说，给定一个大小为 K 的排序，这个排序有对应的$\frac{1}{2}K(K-1)$ 个偏序对。对于表 4-1 中的排序$\mathcal{R}_2$ 的前 5 篇文档也重复相同的操作，我们有

$$[d_{56},d_{123}],\quad [d_{56},d_{84}],\quad [d_{56},d_8],\quad [d_{56},d_6],$$
$$[d_{123},d_{84}],\quad [d_{123},d_8],\quad [d_{123},d_6],$$
$$[d_{84},d_8],\quad [d_{84},d_6],$$
$$[d_8,d_6]$$

比较这两组偏序对，寻找协调和不协调的二元组。让我们把所有协调二元组标为 C，所有不协调二元组标为 D。对于排序$\mathcal{R}_1$，我们有

$$C,D,C,C,$$
$$D,C,C,$$
$$C,C,$$
$$D$$

对于排序$\mathcal{R}_2$，我们有

$$D,D,C,C,$$
$$C,C,C,$$
$$C,C,$$
$$D$$

也就是说，这两个排序一起生成了总共 20 个，即 $K(K-1)$ 个偏序对。在其中，14 对二元组是协调的，6 对二元组是不协调的。那么，这两组排序是协调的概率 $P(\mathcal{R}_1=\mathcal{R}_2)$ 是 14/20，排序间不协调的概率 $P(\mathcal{R}_1\neq\mathcal{R}_2)$ 是 6/20。肯德尔系数被定义为

157

$$\tau(\mathcal{R}_1,\mathcal{R}_2) = P(\mathcal{R}_1 = \mathcal{R}_2) - P(\mathcal{R}_1 \neq \mathcal{R}_2) \tag{4-23}$$

在我们的例子中

$$\tau(\mathcal{R}_1,\mathcal{R}_2) = \frac{14}{20} - \frac{6}{20} = 0.4$$

这表明了两个排序之间具有中等程度的正相关性。

定义 设 $\Delta(\mathcal{R}_1, \mathcal{R}_2)$ 是一个函数，返回两个排序 $\mathcal{R}_1$ 和 $\mathcal{R}_2$ 之间的不协调二元组的个数。那么，这两个排序之间的协调二元组的个数是 $K(K-1)-\Delta(\mathcal{R}_1, \mathcal{R}_2)$。因此，

$$P(\mathcal{R}_1 = \mathcal{R}_2) = \frac{K(K-1) - \Delta(\mathcal{R}_1, \mathcal{R}_2)}{K(K-1)}$$

$$P(\mathcal{R}_1 \neq \mathcal{R}_2) = \frac{\Delta(\mathcal{R}_1, \mathcal{R}_2)}{K(K-1)}$$

肯德尔系数 $\tau(\mathcal{R}_1, \mathcal{R}_2)$ 被定义为 $P(\mathcal{R}_1 = \mathcal{R}_2) - P(\mathcal{R}_1 \neq \mathcal{R}_2)$，即

$$\tau(\mathcal{R}_1, \mathcal{R}_2) = 1 - \frac{2 \times \Delta(\mathcal{R}_1, \mathcal{R}_2)}{K(K-1)} \tag{4-24}$$

注意其值域是 $[-1, 1]$。

对于之前例子中的情况，我们有 $\Delta(\mathcal{R}_1, \mathcal{R}_2)=6$ 和 $K=5$。这样，

$$\tau(\mathcal{R}_1, \mathcal{R}_2) = 1 - \frac{2 \times 6}{5(5-1)} = 0.4$$

该结果与前面的计算结果相同。

肯德尔系数仅是为相同一组元素的不同排序定义的。如果有必要比较不同集合间的排序，那么一种方法是用一个排序中的文档去增广另一个排序中缺少的文档，正如我们在斯皮尔曼系数中所做的那样。

肯德尔系数相比斯皮尔曼系数有更简单的代数结构，有清楚而直观的解释。尽管斯皮尔曼系数更流行，但在计算排序相关性时肯德尔系数其实是更合适的选择。对于一些使用这两个指标的讨论可参考 [678，1208]。

4.4 参考文档集

对于小的文档集，人们可以应用 Cranfield 评价范式，对给定信息需求的检索结果中的每篇文档提供相关性评价。然而，对于大的文档集，不是所有文档都能够针对某个给定的信息需求进行评价。其替代方式是对给定的信息需求，采用由不同的排序算法生成的前 k 篇文档，并把它们集合在一个库（pool）中，仅对这个库中的文档进行评价，这也称为聚合方法（pooling method）。它基于这样的假设：相关文档更有可能在不同的排序顶部找到。这个方法对于包含几百万文档的参考文档集是十分有效的，例如非常流行的 TREC 文档集。我们接下来将对其进行具体讨论。 158

4.4.1 TREC 参考集

信息检索中的研究经常受到两方面的批评。首先，它缺乏一个更形式化的框架作为基础。其次，它缺乏可靠、一致的测试平台和基准。第一项批评是难以回避的，这完全取决于判断文档和信息需求之间相关性的固有主观因素。因此，至少现在，信息检索中的研究不得不在这种非形式化的基础上前进。然而，对于第二项批评，我们可以采取一些措施，即我们接下来讨论的内容。

1. TREC 会议

在信息检索研究开展的 30 年间，其实验都是基于相对较小的测试集，这样的条件无法反映大的文献环境下的主要问题，而且很难在不同检索系统之间做出比较，因为不同的研究小组分别集中在检索的各个不同方面来进行实验，且没有公认的基准。在 20 世纪 90 年代初期，作为对于这种无秩序状态的回应，由美国国防部高级研究计划局（Defense Advanced Research Projects Agency，DARPA）和中央情报局（Central Intelligence Agency，CIA）

资助的 TIPSTER 文本计划启动了。TIPSTER 计划依靠美国海军空间与海战系统中心（Space and Naval Warface Systems Center，SPAWAR）和国家标准与技术研究所（National Institute of Standards and Technology，NIST）的紧密合作来进行联合管理，直至 1998 年项目结题。其主要的成果之一是在 TIPSTER 计划的第一阶段（1991—1994 年）启动了文本检索会议（Text Retrieval Conferences，TREC）。在马里兰州的 NIST，在 Donna Harman 和 Ellen Voorhees 领导下，TREC 会议逐渐走向繁荣，尽管 TIPSTER 计划已经在 1998 年结束了。

TREC 是一个年度会议，旨在对由数百万文档组成的大型测试集进行实验。每届 TREC 会议都会设计一组实验。参加这个会议的研究小组通过这些实验来比较他们的检索系统。关于举办 TREC 会议的目的，在 NIST TREC 网站 [1594] 上可以找到一份清楚的声明，内容如下所示：

> TREC 系列会议是由 NIST 和 DARPA 信息技术办公室作为 TIPSTER 文本计划的一部分进行联合主办的。该系列会议的目的是鼓励在大规模文本应用下的信息检索研究，即提供大规模测试集，统一评价过程，以及为试图比较其评测结果的组织机构提供一个论坛。TREC 会议的与会人员必须是参加了 TREC 检索任务的研究和开发人员，或者是从主办单位挑选出来的政府公职人员。
>
> TREC 会议的参与者采用了广泛的检索技术，包括使用自动同义词典、复杂的
> 159 索引项权重赋值、自然语言技术、相关反馈和高级模式匹配。每个系统在相同的 2GB 文本（超过 100 万篇文档）测试集和一组称为“主题”（topic）的给定信息需求上运行。检索结果用公共评价包来评价，这样不同的研究组可以比较不同技术的效果，并确定这些差别是如何影响系统质量的。

由于文档集是在 TIPSTER 计划下构建的，因此它们经常称为 TIPSTER 或者 TIPSTER/TREC 测试文档集。然而，在这里，为了简单起见，我们称之为 TREC 文档集。

第 1 届 TREC 会议在 1992 年 11 月于 NIST 举办，而第 2 届 TREC 会议举办于 1993 年的 8 月。在 2006 年的 11 月，第 15 届 TREC 会议也在 NIST 举办，共有 107 家参加单位 [1649]。第 18 届 TREC 会议于 2009 年 11 月在 NIST 举办。对于各届 TREC 会议，详情见 [1594]。

接下来，我们将简要地讨论 TREC 文档集和 TREC 会议的（基准）任务。大部分的 TREC 测试集是由三个部分组成的：文档、样例信息需求（在 TREC 术语中称为主题）和每一个样例信息需求的一组相关文档。而且，TREC 会议也包括一组作为基准的任务。

2. 参考文档集

TREC 的主文档集在过去数年中稳步地增长。在 TREC-3 中，文档集的大小大约是 2GB；而在 TREC-6 中，已经增长到大约 5.8GB。在 TREC-15 中，太字节（Terabyte）任务的测试文档集（称为 GOV2）共有从“.gov”域下网站抓取的 2500 万 Web 文档。如何获取测试数据和主题，及其相关文档集的信息可以在 NIST TREC 的网站上找到 [1594]。

TREC-6 Ad Hoc 参考集存储在 5 张 CD-ROM，每张大约有 1GB 的压缩文档。还有一张额外的磁盘包括了分流（routing）任务的数据，但我们在这里不讨论。Disk 1～Disk 5 也用于 TREC-7 和 TREC-8 会议。这些文档出自以下来源：

WSJ——《Wall Street Journal》（华尔街日报）

AP——美联社，新闻专线（Associated Press，news wire）

ZIFF——计算机文章选读，齐夫-戴维斯（Computer Selects articles，Ziff-Davis）
FR——《Federal Register》（联邦公报）
DOE——《US DOE Publications》（美国能源部报告，摘要）
SJMN——《San Jose Mercury News》（圣荷西信使报）
PAT——《US Patents》（美国专利）
FT——《Financial Times》（金融时报）
CR——《Congressional Record》（国会记录）
FBIS——美国中央情报局对外广播情报处（Foreign Broadcast Information Service）
LAT——《LA Times》（洛杉矶时报）

表 4-2 说明了每张 CD 的内容和一些关于文档集的简单统计，来自 [1651]。所有文档加注了 SGML 标记使得解析变得容易。所有文档的主要结构是一个包含文档编号的＜DOC- 160
NO＞域，以及包含文档正文的＜TEXT＞域。子集之间的次结构可能不同，以此保存原始文档中的部分结构。这是 NIST 对于格式的原则：更可能多地保留原来的结构，并提供公用的框架，这样使得文档解析变得简单。

表 4-2 用于 TREC-6 的文档集。依照 [1640]，禁用词没有去掉，也没有抽取词干

磁盘	内容	大小 (Mb)	文档数量	词数/文档 (中位数)	词数/文档 (平均数)
1	WSJ，1987—1989	267	98 732	245	434.0
	AP，1989	254	84 678	446	473.9
	ZIFF	242	75 180	200	473.0
	FR，1989	260	25 960	391	1315.9
	DOE	184	226 087	111	120.4
2	WSJ，1990—1992	242	74 520	301	508.4
	AP，1988	237	79 919	438	468.7
	ZIFF	175	56 920	182	451.9
	FR，1988	209	19 860	396	1378.1
3	SJMN，1991	287	90 257	379	453.0
	AP，1990	237	78 321	451	478.4
	ZIFF	345	161 021	122	295.4
	PAT，1993	243	6 711	4 445	5391.0
4	FT，1991—1994	564	210 158	316	412.7
	FR，1994	395	55 630	588	644.7
	CR，1993	235	27 922	288	1373.5
5	FBIS	470	130 471	322	543.6
	LAT	475	131 896	351	526.5
6	FBIS	490	120 653	348	581.3

TREC 文档的一个例子是在《Wall Street Journal》（华尔街日报）的子集中编号为 880406-0090 的文档，如图 4-11 所示（摘自 [704]）。更多关于 TREC 文档集的详情可从 [1594，1651] 中获悉。

```
<doc>
<docno> WSJ880406-0090 </docno>
<hl> AT&T Unveils Services to Upgrade Phone Networks Under Global Plan </hl>
<author> Janet Guyon (WSJ Staff) </author>
<dateline> New York </dateline>
<text>
American Telephone & Telegraph Co. introduced the first of a new generation of phone
services with broad ...
</text>
</doc>
```

图 4-11 编号为 WSJ880406-0090 的 TREC 文档

3. TREC Web 文档集

Web 检索任务是在 TREC-9 中引入的，旨在构建一个能够最大程度模仿 Web 环境的文档集。其目的是把注意力从在过去的 TREC 会议中占主导的、封闭的随机（ad hoc）检索任务转移到 Web 环境下，在这种环境下，ad hoc 任务中采用的评测方法无法用于数以亿计的查询和文档。

TREC Web 文档集显示在表 4-3 中。这些文档集和任务，以及之前引入的超大规模文档
161 集（Very Large Collection，VLC），对于信息检索的研究有着十分积极的影响。除了 TREC 之外，许多发表在主流会议和期刊中的论文也基于 TREC 开发的 VLC/Web 文档集开展实验。

表 4-3 TREC Web 文档集。VLC2 文档集是从 1997 年的 Internet Archive 中爬取的，WT2g 和 WT10g 文档集是 VLC2 集合的子集。“.GOV”文档集源自于对“.gov”因特网站点的爬取，由 Waterloo 大学在 2002 年完成的。“.GOV2”文档集是 NIST 和 Waterloo 大学在 2004 年的合作成果

文档集	文档数量	平均文档大小	文档集大小
VLC2(WT100g)	18 571 671	5.7KBytes	100GBytes
WT2g	247 491	8.9KBytes	2.1GBytes
WT10g	1 692 096	6.2KBytes	10GBytes
.GOV	1 247 753	15.2KBytes	18GBytes
.GOV2	27million	15KBytes	400GBytes

4. 信息需求主题的例子

TREC 文档集包含一组样例信息需求，可用于测试一个新的排序算法。每个信息需求都是用自然语言进行描述的。在 TREC 术语中，测试信息需求称为主题（topic）。一个信息需求的例子是 TREC-3 中编号为 168 的主题，如图 4-12 所示（源自参考文献 [704]）。

把信息需求（主题）转变为一个系统查询（即一组索引项、布尔表达式、模糊表达式）的任务必须由系统本身完成，作为整个评价过程中的一个重要组成部分。

在前八届 TREC 会议中准备的主题数量累计达到了 450 个。编号 1～150 的主题用于 TREC-1 和 TREC-2 会议。它们是由真实系统中的熟练用户编写的，代表了长期存在的信息需求。编号 151～200 的主题用于 TREC-3 会议，这些主题更短，有着更简单的结构，仅包
162 括三个子域，分别是标题（Title）、描述（Description）和陈述（Narrative），如图 4-12 所示。编号 201～250 的主题用于 TREC-4 会议，相比过去的主题，它的长度更短。TREC-5 包含了主题 251～300，TREC-6 包含了主题 301～350，这些主题是通过和 TREC-3 主题一

样的方式进行准备的，根据 TREC-4 中的主题进行扩展，因为 TREC-4 的主题被认为是太短了。TREC-7 和 TREC-8 各增加了额外的 50 个主题，这样总数达到了 450 个。

<top>

<num> Number: 168
<title> Topic: Financing AMTRAK

<desc> Description:
A document will address the role of the Federal Government in financing the operation of the National Railroad Transportation Corporation (AMTRAK).

<narr> Narrative: A relevant document must provide information on the government's responsibility to make AMTRAK an economically viable entity. It could also discuss the privatization of AMTRAK as an alternative to continuing government subsidies. Documents comparing government subsidies given to air and bus transportation with those provided to AMTRAK would also be relevant.

</top>

图 4-12 TREC 文档集中编号为 168 的主题

5. 每个信息需求的相关文档

在 TREC 会议中，每个样例信息需求的相关文档集是从可能相关的文档库中获取的。这个库是由参与会议的各个检索系统所产生的排序中前 K(通常 $K=100$) 篇文档构成的。然后，将库中的文档提交给评测人员，他们最终决定每篇文档的相关性。

这项估计相关性的技术称为聚合方法（pooling method）[1651]，它基于两个假设：首先，大部分相关文档收集在这个集成库中了。其次，不在库中的文档可以认为是不相关的。这两个假设在 TREC 会议的测试中证明是准确的。这些相关性评估的详细描述可以在 [704，1651] 中找到。

6. TREC 会议的基准任务

TREC 会议包括两个主要的信息检索任务 [704]。第一项称为随机（ad hoc）检索任务，有一组新的（常规）需求，运行在一个固定的文档库上。这是在图书馆系统中经常出现的情况，其中用户对于一组固定的文档询问新的查询。第二项称为分流（routing）任务，即在一个持续变化的文档库上搜索一组固定的需求。这是类似于过滤的任务，对于一组动态 163
的文档（比如，新闻摘要服务）总是询问一组相同的问题。然而，与纯过滤任务不同的是，检出文档必须是排序的。

对于 ad hoc 任务，参加的系统会收到测试信息需求，并在一个预先定义的文档集上执行它们。对于分流任务，参与系统会收到测试信息需求和两个不同的文档集。第一个文档集是用于训练的，从而可以调整检索算法。第二个文档集用于测试调整好的检索算法。

从 TREC-4 会议开始，除了 ad hoc 和分流任务外，引入了新的次一级任务，旨在允许不同系统间进行更特殊的比较。在 TREC-6 中，8 项特别的次一级任务被加入进来，具体如下所示。

- **中文**（Chinese）——文档和主题都是中文的 ad hoc 任务。
- **过滤**（Filtering）——检索算法只需要确定一篇新到的文档是相关（相关的话，就接收这篇文档）还是不相关的（不相关，则丢弃此文档），不需要提供文档的排序。测试数据（输入的文档）是根据时间戳的顺序处理的。
- **交互**（Interactive）——在这项任务中，搜索人员和检索系统互动，以确定相关文

档。文档被裁定为相关或不相关（不提供排序）。

- **自然语言处理**（NLP）——这项任务旨在验证与基于索引项的传统检索算法相比，基于自然语言处理的检索算法是否具有优势。
- **跨语言**（Cross language）——ad hoc 任务，文档使用某种语言，但是主题使用另一种语言。
- **高精度**（High precision）——检索系统的用户被要求在 5 分钟内检出 10 篇能回答一个给定（而且之前是未知的）信息需求的文档。
- **语音文档检索**（Spoken document retrieval）——文档是电台广播新闻的书面转录。旨在激励对于语音文档的检索技术的研究。
- **超大规模语料库**（Very large corpus）——ad hoc 任务，检索系统必须要处理 20GB 规模的文档集（750 万篇文档）。

在 TREC-7 中，NLP 和中文次级任务不再继续开展。此外，由于普遍认为过滤任务是一个更现实的分流任务，分流任务不再作为主要任务。TREC-7 也包括了一项新的任务，称为查询任务（query task），其中为每个样例信息需求生成了多个不同的查询 [1651]。这项
164 任务的主要目的是研究依赖于查询的检索策略在 TREC 文档集上可能会导致的问题，这些问题源于信息需求的稀疏性，而过去的 TREC 会议中的信息需求之间很少有重叠的情况产生。

在 TREC-8 中，信息检索界意识到从 ad hoc 任务中无法学到更多额外的东西。在经过 8 年广泛的实验之后，大家清晰地认识到对于由不同系统所产生的结果，其质量都进入了平台期。尽管排序算法确实在过去数年间有了一些进步，但是在最后的几年里已经趋于平稳，这意味不再有充足的理由去花费大量资源运行 ad hoc 任务了。另外，这也标志着独立研究能够在已经完善的 TREC 参考文档集上继续。因此，ad hoc 任务在 TREC-9 中不再开展。

最近的 TREC 会议集中在新的、还没有完善设立的任务上。其目的在于用这些任务的经验去开发可供学术界进一步研究和实验的新参考文档集。举个例子，在 2006 年举办的 TREC-15 中，主要的任务是问答（query answering）、基因检索（genomics）、太字节检索（terabyte）、企业检索（enterprise）、垃圾过滤（spam）、法律检索（legal）、博客检索（blog）。最近的 TREC 会议中的具体任务和文档集可参考文献 [1649，1650]。

除了提供要执行任务的详细描述之外，TREC 会议也清晰地区分了把以自然语言方式表达的信息需求转化为查询表示（这可能是向量形式和布尔形式）的两种基本技术。在 TREC-6 会议中，可使用的查询构建方法分为自动方法，它从测试信息需求完全自动地产生查询，以及手动方式，即采用除自动方式之外的任何其他方法 [1651]。

7. TREC 会议的评测指标

TREC 会议使用四项基本的评测指标：摘要统计表、平均召回率-精度、文档等级平均精度和平均精度直方图（这些指标的详情可见 4.3 节）。

- **摘要统计表**——这张表总结了对于某给定任务的统计数据。这些统计数据包括：主题（信息需求）的个数、在所有主题上检出的文档数量、所有主题上有效检出的相关文档数量、所有主题上可能检出的相关文档数量。
- **平均召回率-精度**——这是一张表或者图，含有 11 个标准召回率在所有主题上对应的平均精度。由于单个查询的召回率水平很少和标准召回率水平相同，因此通常使用插值来定义标准召回率水平上的精度。而且，也可能包括从所有查询所见过的相关文档中产生的非插值平均精度。

- **文档等级平均精度**——这些是在特殊的文档个数阈值上（而非标准召回率水平）计算的精度数值。举例来说，可能要综合排名前5、10、20、100篇文档处的精度来计算其平均精度。而且，也可以提供在所有查询上的平均R精度（R-precision）。 165
- **平均精度直方图**——这是一张图，其中每一个主题分别对应一个指标。直方图对比了一个给定主题的检索结果的平均精度和所有其他系统对于这个主题给出的检索结果的平均精度，它能让我们具体了解单个系统对于哪些主题能回答得很好，哪些不能提供良好的答案。

4.4.2 其他参考集

除了上述讨论的文档集之外，其他受到关注的参考文档集包括INEX、Reuters、OHSUMED、NewsGroups、NTCIR和CLEF。

1. INEX 参考集

INEX（Initiative for the Evaluation of XML Retrieval）是一项专门为评价XML检索效果而设计的测试数据集。它对于XML界而言是非常重要的，扮演了类似于TREC文档集在信息检索界的角色。在第13章中还会进行详细的讨论。

2. Reuters、OHSUMED 和 NewsGroups 参考集

Reuters是一个由路透社（Reuters）出版的新闻文章构成的参考集。它包含了超过80万篇的文档，分为103个主题类别。OHSUMED是一个由大约348 000份医疗参考文献组成的参考集，这些文献出自于Medline数据库，是从1987—1991年出版的270份期刊中挑选出来的。每篇参考文献都包含了人工赋予的医疗主题词（Medical Subject Headings，MeSH）或者医疗类别。NewsGroups是一个由数以千计的新闻组信息组成的参考集，它分成了20个组，而这些组可以广义地解释为类别。这三个文档集中，每篇文档都有分类（category）或种类（class）信息，这使得它们尤其适合于文本分类算法的评价。因此，在8.6.6节中还会继续讨论它们。

3. NTCIR 参考集

NTCIR(NII Test Collection for IR Systems）项目［1212］每年都会举办名为NTCIR的研讨会，旨在加强信息检索、问题回答、文本摘要和信息抽取等领域的研究。这些研讨会收集了由日文专利和英文专利组成的多种参考集。它们一般称为NTCIR文档集，可以用于在专利检索、专利翻译和跨语言检索上进行详细的实验。举例来说，NTCIR-7 PATMT（Patent Translation Test）数据集［1213］包括了180万个翻译句对（日文-英文)、5200个测试句对、124个查询以及对翻译结果的人工评价。 166

4. CLEF 参考集

跨语言信息检索与评价研讨会（Workshop on Cross-Language IR and Evaluation，CLEF)，是一个主要针对跨语言检索（Cross-Language IR，CLIR）研究及相关问题的年度会议。为了支持实验过程，会议在过去数年间收集了多个不同的CLEF参考集。它们支持不同类型的CLIR实验，例如多语言文档检索、交互式跨语言检索、多语言问题回答、跨语言图像检索、多语言信息过滤、跨语言视频检索、知识产权和日志文件分析。感兴趣的读者可参考文献［393］。

4.4.3 其他小规模测试文档集

在过去几年里，信息检索界也使用了许多小的测试文档集。由于它们的规模较小，因此

不再被视为代表现有水平的测试文档集。尽管有这种局限性，但它们仍然是有用的，对于新的排序算法来说，在利用 Web 和 Trec 等现有水平大规模文档集上进行验证以前，可以先在小文档集上进行早期实验。

一个具有代表性的包含 9 个小规模测试文档集的参考集是由弗吉尼亚理工学院的 Ed Fox 收集的。这些文档集的基本属性在表 4-4 中列出。

表 4-4 用于早期实验的小规模测试文档集

文档集	主题	文档数量	查询数量
ADI	Information Science	82	35
CACM	Computer Science	3200	64
ISI	Library Science	1460	76
CRAN	Aeronautics	1400	225
LISA	Library Science	6004	35
MED	Medicine	1033	30
NLM	Medicine	3078	155
NPL	Elec. Engineering	11 429	100
TIME	Ceneral Articles	423	83

另一个受到关注的小规模测试文档集是囊肿性纤维化（Cystic Fibrosis，CF）文档集 [1454]。它是在 MEDLINE 数据库中由"cystic fibrosis"索引的 1239 篇文档和由一名在囊肿性纤维化方面有着 20 年临床和研究
167 经验的专家生成的 100 个信息需求组成的。与别的数据不同，这个文档集对每篇相关文档分别提供了 4 个相关性分数。3 个相关性分数是由专家提供的，而第 4 个相关性分数是由一个医疗目录学家提供的。分数在 0～2 之间，2 表示高相关性。

尽管其规模小，但 CF 文档集有两个重要特性。首先，其相关分数是由专家通过仔细地评价策略直接生成的。其次，相对于文档集的规模，它包括了多个信息需求，因而各个查询向量之间也有重叠。这使得我们能够测试利用过去的查询会话来提高检索质量的检索策略。

4.5 基于用户的评价

用户的偏好受到用户界面以及与界面交互的难易程度的影响。举例来说，现在众所周知的是，搜索引擎的用户会首先查看结果页面的左上角（见第 5 章）。这样，改变结果页面的布局可能会影响用户的评估和他们的行为。对用户界面和由用户开始的交互过程进行适当的评价需要使用专门的交互系统评价方法，这种方法超出了 Cranfield 实验的框架，我们接下来将进行具体的讨论。

4.5.1 实验室中的人工实验

为了评价用户界面对用户偏好的影响，一般需要运行多个评价会话。除了用户界面的几个专属特性之外，它们看起来几乎相同，以此测量它们对于用户偏好的影响。这类评价方法在实验室封闭的环境中以及仔细挑选的实验人员身上可以完成得更好。

通过实验室中的人工实验可以理解用户和系统交互的动态特性，这是用静态的参考集无法评价的。它们的缺点是需要昂贵的代价来建立或者重复，而且局限于一小组由相对少量的人员提出的信息需求。尽管有这些缺点，但实验室中的人工实验还是很有价值的，因为它对基于参考集的评测所产生的信息进行了补充。

尽管我们在这里不会广泛地讨论实验室中的人工实验，但我们接下来会介绍并排面板——实验室中，利用人来进行评价的一个实例。进一步的参考文献可以在 4.8 节中找到。

4.5.2 并排面板

如果要评价由两个不同系统或者同一个排序函数的不同版本所产生的结果，一种形式是将它们产生的前 K 个检索结果并排显示并评测。典型地，我们会查看两个系统对于给定查

询所产生的前 10 篇结果，它们并排显示，一组在屏幕的左半部分，另一组在屏幕的右半部分。并排显示结果可以对照 1）对于每个主题的意见差异；2）顶部排序结果对于用户意见的影响。我们的讨论基于文献 [1581]。

这里讨论的方法称为并排面板（side-by-side panel），它由如下部分组成：收集两个搜索引擎对于相同查询的排名靠前的答案，并排展示它们，且使用一个共同的背景来遮盖布局当中那些可能会暴露其答案来源的细节。图 4-13 显示了一个例子，这个面板包含了由雅虎和谷歌搜索引擎对于查询“information retrieval evaluation”生成的前 5 项答案。 168

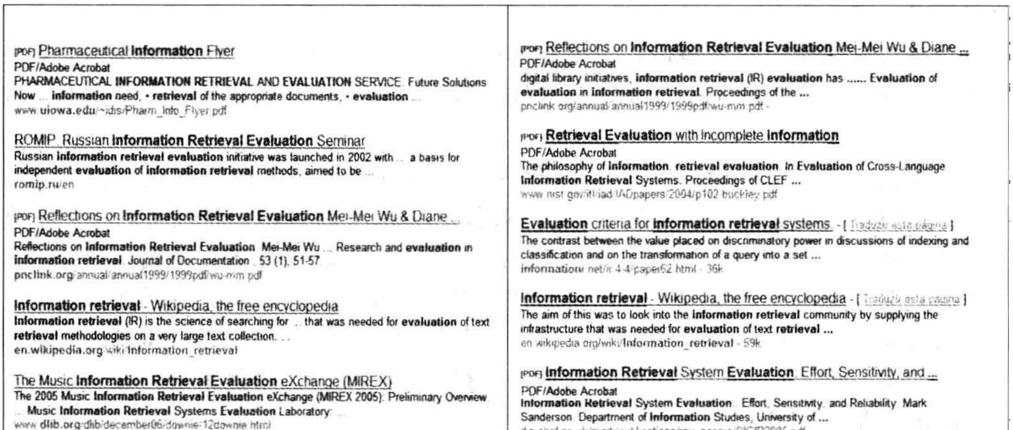


图 4-13 包含了由雅虎和谷歌搜索引擎对于查询“information retrieval evaluation”生成的前 5 项答案的并排面板。布局已经把图标和标记去掉，因为它们可能会透露出答案的来源

并排实验的评价只需要简单地判断哪一边在评估员的眼中能为给定的查询提供更好的结果。举例来说，对于图 4-13 中的查询“information retrieval evaluation”，4 名评估员认为左边的面板更好，而 1 名评估员认为右边更好。这样我们就有理由相信列在左边面板的答案要比右边面板的答案和查询更相关。注意这里并没有评价某个特定答案的优劣。

通过记录选中的用户和这两个面板的交互过程，我们就能推断用户更偏向哪组答案集作为查询的答案。由于我们能即时生成并排面板（通过包装实时查询流的结果），因此它们可以用于快速比较不同的搜索引擎。而且，由于其动态的特性，即它们可以在实时搜索流上执行和度量的这一特性，因此它们在评价受关注的一段时间内的短期趋势方面具有吸引力。

在并排实验中，用户明白他们正在参加一项实验。而且，并排实验不能在之前执行过的相同条件下重复，因为实时查询流的条件会根据时间不停地变化。最终，并排面板无法测量系统 A 比系统 B 好多少，也不能直接在多个系统之间进行比较。尽管有这些缺点，但并排面板构成了一种动态评价方法，提供对系统深入的理解，从而对其他评测方法进行了有效的补充。

4.5.3 A/B 测试

A/B 测试，也称为木桶测试（bucket testing），是如今的一种流行方法。它是由数以千计的网站预选用户来执行评价实验的。举例来说，它可能会向预选用户展示页面布局的改变。通过分析用户如何对这个变化做出反应，可以获知这个修改是否是积极有益的。即使不是实验室的环境，但它也提供了一种人工实验的方法。 169

通常，被选中的用户组只占网站全体用户的一小部分，但却非常具有代表性。这个技术对经常被使用的站点是尤其重要的，因为启用一个糟糕的修改可能会让许多用户感到不快。更多

关于 A/B 测试的讨论，请参考文献［920，921，456］和 2.5 节的搜索界面设计与评价。

4.5.4 众包

目前，相关性评价方法有许多的局限性。举个例子，大量的编辑人员使用 Cranfield 范式评价搜索结果的相关性是代价昂贵的，很难规模化。在没有这些编辑人员的情况下，学术研究人员经常依靠学生志愿者组成的小组。由于学生有限，且他们的时间也受限制，因此测试集合通常比期望的要小，使得我们难以测试待评估实验系统的检索质量是否有统计意义上的显著差异。

行为数据要比编辑数据更廉价，但也有其局限性。它需要获得大量的数据流，而研究人员在测试实验系统的时候并不一定能得到这样的数据，而且这样的流数据对某些任务是没有用处的。

近来，众包（Crowdsourcing）已经演化为一种可行的相关性评价方法，因为它在更大规模数据上保持了编辑方法的灵活性［37］。众包这个词用来描述某种任务，它们外包给一大组称为“工作者（worker）”的人，而不是由正式雇员或者合同工来完成。众包公开号召人们来解决问题，或者执行某项任务，通常使用者需要支付费用来得到这项服务。

较低的实验费用使得这个方法很有吸引力，并能快速地测试新想法。世界各地有越来越多的网络用户参与进来，这对于实验来说是一个很好的样本集。举例来说，众包已经用来验证搜索片段的质量，详见第 11 章。

使用众包执行评价的最重要因素之一，是仔细地设计实验。要用简单有效的方法来提出正确的问题，这需要在问卷和调查的设计过程中有正确的方针和准则。工作者和 Web 浏览器进行交互，所以使用众所周知的可用技术来展示信息是很重要的。工作者不是信息检索专家，所以任务设计人员应该提供清楚的指示，表明如何评价文档的相关性，如果可能的话，应当提供样例。

从工作者那里获得反馈信息是一种极好的验证答案的机制，在一切实验中都应该具备。对反馈的后处理能有助于改进指示和系统。

之前的研究已经表明，这种方法能为 TREC 数据［36］和自然语言文本标注［1499］
170 产生相同质量的评价结果。其他的应用是用于机器学习的数据生成和排序函数的特征选择。应使用测试者之间的一致性指标来确保这些评价是良好一致的［888］，无论平台是内部还是外部系统。对于外部的众包平台，AMT 是最为著名的，我们接下来将进行具体的讨论。

亚马逊土耳其机器人

亚马逊土耳其机器人（Amazon Mechanical Turk，AMT）是众包平台的一个例子［40］。在这项服务中，称为“turker”的参与人员执行称为“HIT”的人类智能任务，以换取少量的报酬。任务是由评价需求方提出的。尽管需求方无法知道参与者的身份，但这项服务仍然能产生高质量的结果［861，1540］。

在 AMT 上，需求方是有工作需要完成的个人或组织。“turker”是希望注册来完成任务的人，在系统中描述为工作者。每一个人类智能任务（Human Intelligence Task，HIT），或者要执行的工作单元，都有一个相对应的支付报酬，以及分配的完成时间；工作者在选择是否执行任务前，能看到样例 HIT 以及支付的报酬和时间信息。可以使用资格测试来控制工作的质量。资格测试是一组问题（类似于 HIT），工作人员必须回答来获取资格然后才能在这些任务上工作。

AMT 使得开发人员能够把人类智能作为他们应用程序的核心部分。开发人员使用 Web

服务 API 提交任务，批准已完成的任务，把答案集成到他们的软件应用中。对于应用，这项工作看起来非常像某种远程调用。应用发送需求，服务返回结果。人们访问网站来寻求任务，为他们完成的工作收取报酬。除了 API 之外，也可以在原型实验中使用包含多个有用特性的控制面板来进行互动。

4.5.5　使用点击数据的评价

尽管参考集提供了有效的手段去评价结果集合的相关性，但它们往往只能应用在相对少量的查询上。举例来说，Web 搜索引擎的查询日志通常是由数十亿查询构成的，而评估人员的评价结果就仅限于由几百个查询组成的查询样本，要超出这个阈值是非常昂贵的。因此，基于参考集的 Web 搜索引擎评价有着很大的局限性。搜索引擎的动态特性，即用户偏好和文档集的经常变化，使得这些局限性变得更加棘手。也就是说，Cranfield 范式不是很适合直接应用在 Web 中，需要由别的评价方法来补充。

一种非常有希望的方法是基于点击数据的分析进行评价。这是通过观察用户点击某个给定文档（当该文档出现在给定查询的答案集中）的频繁程度来实现的。这一方法非常有吸引力，因为点击数据能以很低的代价来搜集，而不需要额外的用户开支。为了保证该方法的有效性，需要进行仔细的实验设计，因为点击数据组成了一种隐式的用户反馈，这比由评估人员提供的直接相关反馈有更多的噪声。我们这里的讨论基于 Joachims [842] 的工作。 171

1. 有偏的点击数据

一种对点击数据的直接使用方法是比较两个不同的搜索引擎 A 和 B，所生成的排序$\mathcal{R}_A$和$\mathcal{R}_B$ 的点击率。举例来说，假设不同的用户在不同的时间指定了相同的查询。对于每一个这样的查询，我们随机选择两个搜索引擎中的一个，并把结果展示给用户，而不让他们知道使用了哪个引擎。然后，我们记录用户对答案的点击。这个过程在多个不同的查询上重复，并对每个查询记录其点击数据，这样我们就可以对每个查询做平均了。

通过比较数以百万计的点击数据，我们希望能判断用户群更偏向于哪个搜索引擎。然而，点击数据难于解释。举例来说，假设有查询 q，用户已点击了排序$\mathcal{R}_A$ 的答案 2、3 和 4，以及排序$\mathcal{R}_B$ 的答案 1 和 5。对第一种情况，平均点击排序位置是 $(2+3+4)/3$，等于 3。在第二种情况下，是 $(1+5)/2$，也等于 3。这是否意味着对这个查询，两个搜索引擎提供了差不多的结果？而引擎 A 的用户点击了首篇文档下面的文档，是否这个情况更重要呢？这个例子说明了点击数据难以分析。

上面例子的主要问题是点击数据不能完全指示相关性。也就是说，被高度点击的文档不一定是相关的。它们可能只是相对于答案集中的其他文档更符合用户需求而已。因此，由于点击数据是一个相对测度，因此难以用它们来直接比较两个不同的排序算法，因为一个算法产生的结果和另一个没有关联。

我们说收集到的点击数据是有偏的，因此不能用于直接比较搜索引擎。替代的方法是融合两个排序来收集无偏的点击数据。

2. 无偏的点击数据

为了从用户处搜集无偏的点击数据，我们混合两个排序算法的结果集，从而保证任一排序产生的文档点击数据总是和另一排序相关联，这样我们就可以比较两个排序的点击数据。为了混合两个排序的结果，我们查看并混合每个排序靠前的文档结果，以保证在结果的答案集中没有重复的文档。在图 4-14 中显示的算法能达到这种效果，它是在文献 [842] 中提出的。调用 $combine_ranking(\mathcal{R}_A, \mathcal{R}_B, 0, 0, \varnothing)$，把排序$\mathcal{R}_A$ 的首篇文档插入融合排序$\mathcal{R}$。

后面再调用 *combine_ranking*($\mathcal{R}_A$，$\mathcal{R}_B$，1，0，$\mathcal{R}$)，把$\mathcal{R}_B$ 中的一篇文档放入了融合排序$\mathcal{R}$中。唯一的限制是从$\mathcal{R}_B$ 中选出放入的文档应当还没有加入到融合排序$\mathcal{R}$中。为此，我们遍历$\mathcal{R}_B$ 直到我们找到还没有加入到$\mathcal{R}$中的文档，这是通过递归调用函数 *combine_ranking* 来实现的。

```
Input: R_A = (a_1, a_2, ...), R_B = (b_1, b_2, ...).
Output: a combined ranking R.
combine_ranking(R_A, R_B, k_a, k_b, R) {
  if (k_a = k_b) {
    if (R_A[k_a + 1] ∉ R) { R := R + R_A[k_a + 1] }
    combine_ranking(R_A, R_B, k_a + 1, k_b, R)
  } else {
    if (R_B[k_b + 1] ∉ R) { R := R + R_B[k_b + 1] }
    combine_ranking(R_A, R_B, k_a, k_b + 1, R)
}}
```

图 4-14 混合两个排序的算法

融合排序有一个特殊的属性，在前 r 篇答案组成的集合中，来自两个排序的答案数量之差不会超过 1。通过为融合排序搜集点击数据，我们确保数据是无偏的，反映用户可预测的偏
172 好，具体如下所示。

> **无偏的点击数据**：设$\mathcal{R}_A$ 和$\mathcal{R}_B$ 是由两个不同的排序函数生成的结果集合。结果集合是通过调用 *combine_ranking*($\mathcal{R}_A$，$\mathcal{R}_B$，0，0，$\varnothing$) 来确保在排序中的任何一点，前 r 篇文档结果包含$\mathcal{R}_A$ 的前 r_a 个结果以及$\mathcal{R}_B$ 的前 r_b 个结果，且 $|r_a - r_b| \leqslant 1$。融合排序接下来展现给用户，而点击数据也会被记录。

在这些条件下，可以证明：

> 排序$\mathcal{R}_A$ 比排序$\mathcal{R}_B$ 包含更多的相关文档，当且仅当$\mathcal{R}_A$ 的点击率比$\mathcal{R}_B$ 的点击率高。最重要的是，在一般的假设下，如果要比较两个排序算法，基于融合排序点击数据的方法和基于人工评估相关性的方法所产生的比较结果是一致的。

这是一个惊人的结果，它表明了用户点击数据和答案相关性之间的联系。

4.6 实践说明

本章的大部分内容都是度量检索系统对于给定的一组查询在特定的文档集上的平均质量。在实践中，这种标准评价方法有两个需要注意的地方：简单的实验设计和简单的比较指标。

简单的实验设计有助于其他人进行重复验证。也就是说，我们更推荐使用公开的参考集。另一方面，有些试验结果只在一部分文档集上有效，而无法推广到其他文档集上。因此，使用多个文档集可以更好地展示系统的改进程度。对查询也是如此。也就是说，可能会有一个检索系统 A 对于一组查询能获得更好的检索质量，但是对于另一组查询则不如检索系统 B。因此，使用多组尽可能真实的查询，可以更有力地表明我们获得了更好的检索质
173 量。这里另一个问题是，我们应该使用多少查询来确认性能提升的统计显著性。这依赖于使用的统计显著性测度，统计表可以告诉你需要多少实验来达到你想要的显著性水平。遗憾的是，许多研究结果使用任意的、小规模的查询，而没有分析其在某个置信度水平上的有效性。对于这个问题，我们在 4.5.3 节的 A/B 测试中已经有所阐述。另一方面，其他一些作者相信显著性测试不是很有用 [1791]。最后，时间也是一个值得注意的问题，由于文档集和查询可能一直在变化，特别是对 Web 来说更是如此。所以，今天最好的检索系统，不一定在明天仍然是最好的。

第二个需要注意的是比较的方法。首先，和一个简单的基准进行比较会更容易，但是这并不表示你的检索系统打败了最先进的系统。所以，我们鼓励使用更加强大的系统作为基准。另一方面，有些新技术也许无法提高现有水平，但却可能有着不可忽视的其他优点。举例来说，如果一项新颖而简单的技术在性能上能够接近于那些复杂的最优技术，那么它一定

是很引人关注的。其次，仅仅测量平均值可能会给人误导性的结果。举例来说，系统 A 可能在平均水平上比系统 B 要好，但这可能只是由于系统 B 中某个查询的效果很糟糕，而其他查询的结果其实却比 A 更好。所以，检索质量的可变性也是很重要的（参考文献 [777] 中使用了波动性（volatility）这个词）。举例来说，如果有最低检索质量阈值的限制，那么你应当使用一直处于阈值之上的系统 A，而不是虽然平均水平更好，但一半以上的时候都会低于阈值的系统 B。相反的，如果没有系统具有高于阈值的平均检索质量，那么我们会倾向于使用一半时间要高于阈值的系统 A，尽管系统 B 就平均水平而言会更好。

4.7 趋势和研究问题

如今的一个主要趋势是研究交互式用户界面及其评价。其动机源于一个普遍的认识，即有效的检索非常依赖于从用户处获得合适的反馈。现在的主要问题围绕在讨论哪种评价指标最适用于这种情景。近来一些较好的范例是在 4.5 节中已讨论过的并排面板和使用点击数据的搜索日志分析。

另一个重要的趋势是众包，即让 Web 用户来执行定义明确的评价任务，同时只需要较少的支出。它以低廉的代价获得了可扩展性，并可以产生不错的结果（如在 4.5 节中讨论的那样）。如果要对交互频繁的 Web 应用进行仿真和评价，那么就必须采用像众包那样具有良好扩展性的评价方法，考虑到这一点，我们预计未来会看到这个领域更多的活动和研究。

此外，对于精度和召回率的替代指标，它们的提出、研究和特性描述始终受到大家的关注，尤其是在那些精度-召回率数值无法完全涵盖的情景下。两个非常好的实例是 4.3.4 节讨论的 DCG 指标和 4.3.5 节讨论的 *Bpref* 指标。

4.8 文献讨论

Cranfield 评价范式，从最初的概念到最新的评价方法，都可以通过 Cleverdon 关于这个主题的许多文章 [395，397，396，399] 来深入了解。Harter 和 Hert 的工作 [709] 讨论 174
了 Cranfield 模型以及对原模型的扩充和变体，总结了涵盖 396 篇参考文献的研究历程。Saracevik 在文献 [1424] 中对评价信息检索系统的方法和过程进行了分析，这个工作非常重要，甚至可以说是历史性的。Buckley 和 Voorhees 在文献 [291] 中提供了关于不同的信息检索评价指标准确性的分析，以及对于在信息检索评价中经常采用的经验法则的讨论（例如用于实验的查询的个数）。

在 Salton 和 McGill 的书中 [1414] 有关于检索评价的章节，写得十分出色。即使已经过时了，但它仍然是值得阅读的材料。对于信息检索评价方法一般意义上的讨论出现在文献 [1372] 中。Khorfage 的书中 [931] 也包含了一章关于检索评价的内容。Mizzaro 的工作 [1145] 讨论了 160 篇论文中的结果，提供了多年来相关性研究的深度总结。

Shaw、Burgin 和 Howel 的论文 [1455，1456] 讨论了测试集的标准和评价方法，并且分别讨论了基于聚类的检索模型和基于向量的检索模型这两种情形。这些论文也讨论了以精度和召回率的调和平均作为关于精度和召回率的单一指标的优势。Raghavan、Bollmann 和 Jung 的论文 [1325，1326] 讨论了对于需要文档弱排序关系的系统，精度和召回率可能存在的问题。Tague-Sutcliffe [1552] 提出了一种信息性测度，用来评测用户在会话中的交互性。Van der Weide、Huibers 和 van Bommel 在文献 [1622] 中提出了和“采摘”（berry picking）模型类似的新模型。

我们对如今信息检索实验中最常用的文档集——TREC 文档集的讨论，是基于 Harman

[704] 以及 Voorhees 和 Harman[1651] 的论文。Voorhees 也提供了把 Cranfield 范式应用到 TREC 文档集中的详细讨论 [1648]。而且，在文献 [1647] 中，她讨论了在 TREC 结果评价时所用的相关性评估方法的改变，以及其所带来的影响。

我们对于其他参考集也进行过讨论，包括 INEX、Reuters、OHSUMED、NTCIR 和 CLEF 文档集。它们涵盖了近年来对结构化文本检索、文本分类、跨语言检索和垂直文档集的研究趋势。更多相关文献的详情见第 8 章和第 13 章。

近年来，关于检索评价的综述主要有两个：一个是 Sanderson[1422] 关于参考集使用情况的总结，另一个是 Kelly[894] 关于交互式信息检索的讨论。

我们对排序相关性测度的讨论涵盖了两种最流行的方法，也就是斯皮尔曼（Spearman）[1052，1511] 和肯德尔（Kendal Tau）[3，898，1208] 系数。我们关于 DCG（折扣累计增益）的讨论基于文献 [828，892]。文献 [892] 对二值相关性系统和分级相关性系统（如 DCG）在 TREC 文档集上进行了比较。在文献 [24] 中分别使用了精度值和 DCG 数值对 Web 搜索结果进行了评价。结论表明用户的满意程度和精度值与 CG 数值都紧密关联，但与 NDCG 数值关系不明显。我们对 BPref 指标的讨论基于文献 [292]。

我们对于 Web 检索评价的讨论涵盖了并排面板 [1581]、基于点击的评价 [842] 和众包 [37，36，1499，888]。尽管这些研究大多数都是最近才提出的，但在这些主题上不停增长的活动和研究兴趣表明，这些是 Web 中重要的问题。

文献 [1163] 讨论了使用信息检索指标来评价基于内容的图像检索（Content-Based Image Retrieval，CBIR）系统的方法。Koenemann 和 Belkin 在文献 [918] 中讨论了关于交互
175 会话的评价，并带有案例研究。最近，Borlund 提出了另一种方法来评价交互系统 [237]。

除了我们已经讨论的这些指标外，还有其他的指标值得关注。其中包括*期望搜索长度*（expected search length），该指标适用于弱排序文档集合；*满意度*（satisfaction），该指标仅考虑了相关文档；*挫折度*（frustration），该指标仅考虑了不相关文档 [931]。在文献
176 [796] 中讨论了如何使用统计测试来评估不同信息检索评价指标的充分性。

相关反馈与查询扩展

5.1 介绍

在没有充分了解文档集的情况下，大部分用户觉得难于提出专为检索而设计的查询。举例来说，搜索引擎用户经常需要重构他们的查询以获得他们所感兴趣的更好结果。这种困难表明，初始查询表式应该看做是初次检索相关信息的尝试，并且可以写出更好的新查询表式来检索出其他有用的文档。

在本章中，我们将考察各种提升初始查询表式的方法，它们大都使用和查询意图相关的信息。我们这里所说的"相关的"信息是指，能用于检出可能和初始查询相关的文档的信息。对于修改查询的过程，如果用户清楚地提供了查询相关文档的信息，那么在文献中通常称为相关反馈（relevance feedback）；如果查询的相关信息用来扩展查询，则称为查询扩展（query expansion）。这里，我们把这两者都称为反馈方法。

我们的讨论不是旨在提供一个关于反馈方法的详尽综述。相反地，它涵盖了经过挑选的文献，我们相信这些文献足够宽泛，可以给出主要问题的概览以及在反馈循环中涉及的权衡问题。我们区别两种基本的方法：1）显式反馈，用于查询重构的信息是直接由用户提供的；2）隐式反馈，用于查询重构的信息是由系统潜在地提供的。我们的讨论自然地导出了一个对反馈方法的分类框架。 177

5.2 反馈方法的框架

相关反馈（relvance feedback）最初由［1375］提出，是指一种反馈循环，其中和当前查询 q 相关的已知文档用来把查询转换为改进查询 q_m，期望查询 q_m 可以返回更多与 q 相关的文档。文献中的经验证据表明相关反馈是有效的，它确实能产生更好的结果，不仅对于文本，对于图像也是如此［290，701，1396，1411，1784］。

然而，获得和当前查询相关文档的信息是昂贵的，需要用户直接干预。举例来说，尽管信息检索系统可以询问用户对于某个给定查询的前10篇文档是否真的相关，但是大部分用户都不愿意提供这样的信息，在Web中尤其是这样。由于其高昂的代价，相关反馈的想法这些年来已经放松了条件，允许使用认为是和查询相关的信息。举例来说，不是去询问用户关于相关文档的情况，而是利用他们已经点击的文档，或者查看结果集中最靠前文档中的索引项。在这两种情况下，如果假设搜集到的信息和原始查询相关，那么我们期望反馈循环会产生更高质量的结果。

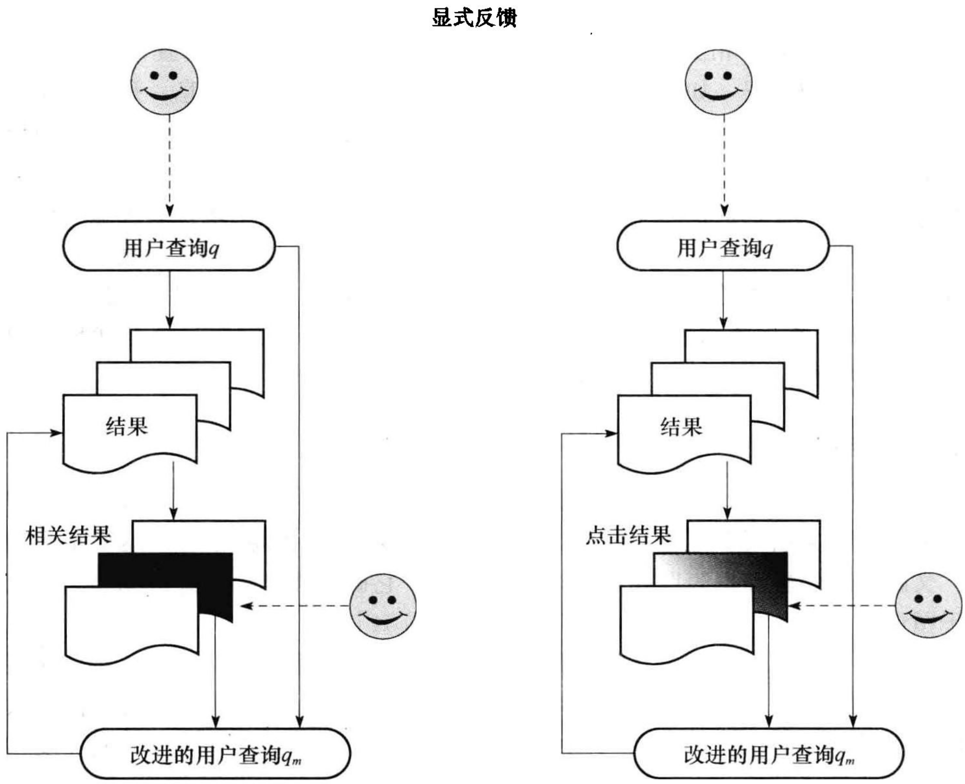
反馈循环是由两个基本步骤组成的：1）判断反馈信息是否和原始查询 q 相关，或者可以看做是相关的；2）如何有效地考虑这些信息，以转化查询 q。步骤1）可以用两种不同的方法来完成：显式地从用户处获得反馈信息，或者隐式地从查询结果或者同义词典等外部资源获得反馈信息。在生成反馈信息后，步骤2）可以通过各种方法来完成，我们在本章中将详细进行讨论。用来收集反馈信息的方法，显式或者隐式，是相关反馈方法的主要区别。

1. 显式反馈信息

在显式相关反馈循环中，反馈信息是直接由用户或者一群人工评估员提供。在其最初的

形成过程中，用户检查排名靠前的文档，标出确实和查询相关的文档。为了最小化误判，反馈信息可以从不同的用户处收集，仅考虑大部分用户赞成的信息。由于用户可能不愿意合作，或者在相关性判断中不可靠，可以考虑的一种方法是邀请多名专家进行相关性评估。但是，无论是何种情况，收集反馈信息都十分昂贵、耗时。

在 Web 中，用户对搜索结果的点击形成了一个新的反馈信息源。一个点击不一定表明一篇文档和查询项是相关的，但是它表明了文档在当前查询背景下用户会感兴趣。图 5-1 说明了两种类型的显式反馈循环：由用户选择的相关结果和由用户点击的结果。在两种情况下，我们注意到用户在反馈循环中都是直接参与的。然而，在第二种情况下，这种参与对用户而言更自然，因为用户不需要偏离当前的任务。而且，点击信息可以在不干扰用户的情况
178 下大量收集，而相关结果却不是这种情况。5.3 节讨论了 Rocchio 相关反馈方法，这是使用相关结果来提高用户查询的经典方法。5.4 节讨论了一个最新提出的解释点击结果的模型。



a）相关反馈　　　　b）点击反馈

图 5-1 显式反馈信息：a）用户选择的相关结果；b）用户点击的结果

2. 隐式反馈信息

在隐式反馈循环中，用户没有参与到反馈过程中，反馈信息隐式地出自系统。有两种基本的方法可以得到隐式反馈信息：1）从结果集合排名靠前的文档中产生反馈信息，这通常称为*局部分析*（local analysis）；2）从外部资源产生反馈信息，例如同义词典，或者从文档集构建的索引项关系等，这通常称为*全局分析*（global analysis）。

图 5-2 说明了这里阐述的两种隐式反馈循环：局部分析和全局分析。在这两种情况下，
179 用户都没有直接参与到反馈循环中来。显然，反馈信息不一定和当前查询相关，这使得如何

应用这些信息变得比由用户显式提供反馈更具挑战性。尽管如此，由于隐式信息丰富，可以以低廉的代价收集，所以人们一直关注使用隐式信息来提高查询结果的方法。5.5 节中讨论了基于局部分析的相关反馈方法。5.6 节讨论了基于全局分析的相关反馈方法。

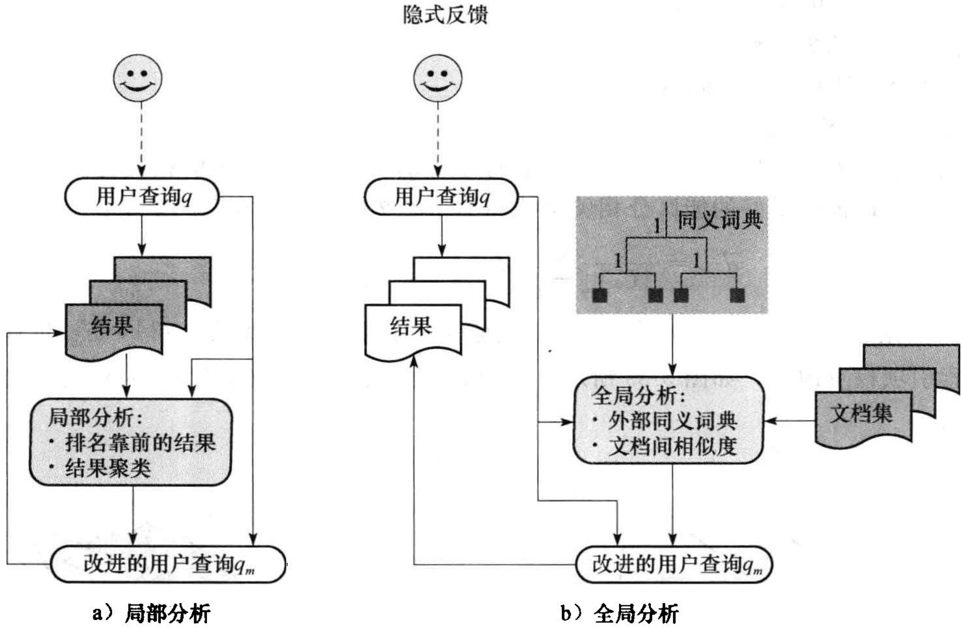


图 5-2 隐式反馈信息：a）局部分析，即从排名最靠前的排序结果中生成反馈信息；b）全局分析，即从外部资源获得反馈信息

5.3 显式相关反馈

在经典的相关反馈循环中，给用户一些检出文档，用户检查并且标出相关的文档。在实际应用中，只有排在前 10 篇（或 20 篇）的文档需要检查。其主要想法是从用户标注为相关的文档中选出重要的索引项，并在新的查询表式中加强这些项的重要性。期望的效果是新的查询可以靠近相关文档，远离不相关文档。

早期使用 Smart 系统 [1408] 的实验以及后来使用概率模型 [1365] 的实验已经证明，使用相关反馈可以提高小规模测试文档集的精度。这样的提高来自于采用了两项技术：添加从相关文档中抽取的新查询项的查询扩展，和基于用户相关性判断的查询项权重调整。 180

相关反馈具有如下的特性：1）它不让用户见到查询重构过程的细节，因为用户需要提供的所有信息只是对文档的评判；2）它把整个搜索任务分段为一系列可以更容易掌控的小步骤。

接下来，我们讨论相关反馈的应用：1）使用向量模型的扩展查询；2）在概率模型下重新调整查询项的权重。我们也将讨论如何评价相关反馈循环。

5.3.1 向量模型的相关反馈：Rocchio 方法

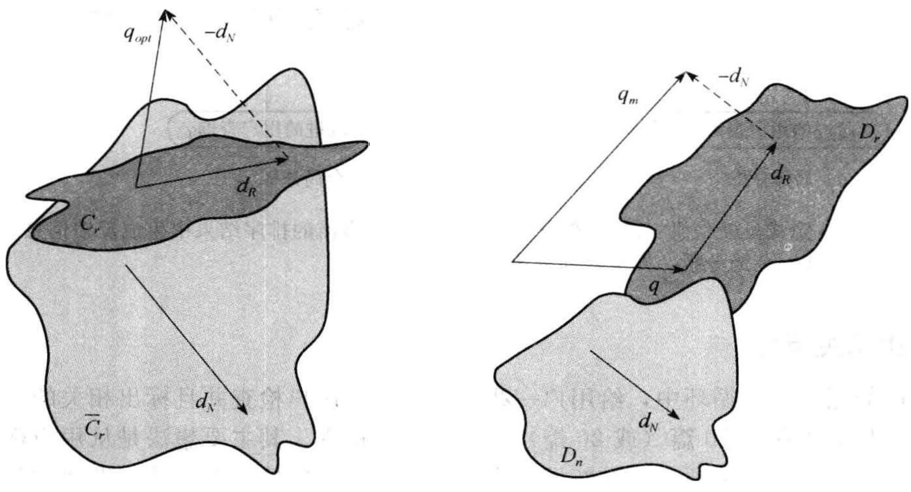
向量模型的相关反馈假设（针对某个查询）的相关文档之间是类似的，即相关文档会彼此聚合。而且，假定不相关文档的索引项权重向量和相关文档的向量相差甚远。主要的想法是重构查询使得它更靠近向量空间中相关文档的区域，远离不相关文档的区域。在此之前，让我们先对处理某个给定查询 q 定义术语，如下所示：

D_r：检出的文档中相关文档的集合；
N_r：集合 D_r 内文档的数量；
D_n：检出的文档中不相关文档的集合；
N_n：集合 D_n 内文档的数量；
C_r：整个文档集中相关文档的集合；
N：整个文档集中文档的数量；
α，β，γ：调整参数。

首先考虑一种乐观情况，其中对于给定的查询 q，事先已知完整的相关文档集 C_r。在这种情况下，可以证明最好的能区分相关文档和不相关文档的最佳查询向量是：

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\forall \vec{d}_j \in C_r} \vec{d}_j - \frac{1}{N - |C_r|} \sum_{\forall \vec{d}_j \notin C_r} \vec{d}_j \tag{5-1}$$

其中 $|C_r|$ 是集合 C_r 的大小，$\vec{d}_j$ 是文档 d_j 的索引项权重向量（见第3章），$\vec{q}_{opt}$ 是查询 q 的最优化索引项权重向量，如图5-3a所示。这个式子的问题是组成集合 C_r 的相关文档是事先不知道的。



a)　　　　b)

图 5-3 Rocchio 反馈过程：a）在已知完整的相关反馈信息（$\vec{d}_R = \sum_{\vec{d}_j \in C_r} \vec{d}_j$）和不相关反馈信息（$\vec{d}_N = \sum_{\vec{d}_j \notin C_r} \vec{d}_j$）的前提下，最优的查询表式；b）Rocchio 查询重构过程，其中初始查询 $\vec{q}$ 被转换为基于检出文档中的相关文档（$\vec{d}_R = \sum_{\vec{d}_j \in D_r} \vec{d}_j$）和不相关文档（$\vec{d}_N = \sum_{\vec{d}_j \in D_n} \vec{d}_j$）构成的改进查询 $\vec{q}_m$

为了避免这一问题，自然的做法是构造一个初始查询，逐渐改变这个最初的查询向量。这个增量式的改变是通过把计算限制在（根据用户的评价）当时已知的相关文档上。有三种
181 相似的经典方法计算改进查询 $\vec{q}_m$，如下所示：

$$Standard_Rocchio: \quad \vec{q}_m = \alpha \vec{q} + \frac{\beta}{N_r} \sum_{\forall \vec{d}_j \in D_r} \vec{d}_j - \frac{\gamma}{N_n} \sum_{\forall \vec{d}_j \in D_n} \vec{d}_j \tag{5-2}$$

$$Ide_Regular: \quad \vec{q}_m = \alpha \vec{q} + \beta \sum_{\forall \vec{d}_j \in D_r} \vec{d}_j - \gamma \sum_{\forall \vec{d}_j \in D_n} \vec{d}_j \tag{5-3}$$

$$Ide_Dec_Hi: \quad \vec{q}_m = \alpha \vec{q} + \beta \sum_{\forall \vec{d}_j \in D_r} \vec{d}_j - \gamma\, max_rank(D_n) \tag{5-4}$$

其中 $max_rank(D_n)$ 是指排序最高的不相关文档。在原始表式中，Rocchio 固定了 $\alpha=1$[1375]，Ide 固定了 $\alpha=\beta=\gamma=1$[806]。上面的表式是最新的变体。当前的理解是这三项技术能产生出类似的结果（在过去，认为 Ide Dec-Hi 略好一点）。

Rocchio 表式基本上是由公式 5-1 直接转化过来的，其中原始查询项添加了进来，如图 5-3b所示。其动机是原始查询 q 包含了重要的信息，这些信息决定哪些文档是相关的。此外，因为包含在相关文档中的信息比不相关文档中的信息更重要 [1414]，常数 γ 应该比常数 β 小。另一种方法是设置 γ 为 0，这样可产生一个正（positive）反馈策略。 182

上述相关反馈技术的主要优点是既简单又可以得到好的结果。简单是由于索引项权重的修改是直接从相关文档集计算得到的。结果好是通过实验观察得到的，这是由于改进的查询向量反映了一部分查询语义。

5.3.2 概率模型的相关反馈

概率模型根据概率排序原则动态地对查询 q 的类似文档进行排序。由 3.2.7 节，我们可以知道文档 d_j 和查询 q 在概率模型中的相似度可以表示为：

$$sim(d_j,q) = \sum_{k_i \in q \wedge k_i \in d_j} \log\left(\frac{P(k_i|R)}{1-P(k_i|R)}\right) + \log\left(\frac{1-P(k_i|\overline{R})}{P(k_i|\overline{R})}\right) \tag{5-5}$$

其中 $P(k_i|R)$ 表示在相关文档集 R 中观察到索引项 k_i 的概率，$P(k_i|\overline{R})$ 表示在不相关文档集合 $\overline{R}$ 中观察到索引项 k_i 的概率。最初，不能使用式（5-5），因为概率 $P(k_i|R)$ 和 $P(k_i|\overline{R})$ 是未知的。第 3 章已经讨论了许多自动估计这些概率的方法（即在没有用户反馈的情况下）。在有用户反馈信息的情况下，这些概率是以一种略有不同的方法估计的。

对于初始搜索（在没有检出文档的时候），通常做出的假设包括：1）$P(k_i|R)$ 对所有的索引项 k_i 是常数（典型的是 0.5）；2）索引项概率分布 $P(k_i|\overline{R})$ 可以近似为整个文档集的分布。由这两个假设可得：

$$P(k_i|R) = 0.5$$

$$P(k_i|\overline{R}) = \frac{n_i}{N}$$

如前所述，n_i 表示文档集中包含索引项 k_i 文档的数量。把它代入式（5-5）中，可以得到：

$$sim_{initial}(d_j,q) = \sum_{k_i \in q \wedge k_i \in d_j} \log\left(\frac{N-n_i}{n_i}\right)$$

对于反馈式搜索，由先前检出的相关或不相关文档对应的累积统计量，用来估计概率 $P(k_i|R)$ 和$P(k_i|\overline{R})$。假设 $n_{r,i}$是集合 D_r 中包含索引项 k_i 的文档数量。那么概率 $P(k_i|R)$ 和 $P(k_i|\overline{R})$ 可以近似为：

$$P(k_i|R) = \frac{n_{r,i}}{N_r}$$

$$P(k_i|\overline{R}) = \frac{n_i - n_{r,i}}{N - N_r} \tag{5-6}$$

使用这些近似，式（5-5）可以重写为：

$$sim(d_j,q) = \sum_{k_i \in q \wedge k_i \in d_j} \log\left(\frac{n_{r,i}}{N_r - n_{r,i}}\right) + \log\left(\frac{N-N_r-(n_i-n_{r,i})}{n_i-n_{r,i}}\right)$$ 183

值得注意的是，这里与向量空间模型不同，并没有对查询进行扩展。而是对相同的查询

项使用了由用户提供的反馈信息来重新调整。

式（5-6）会出现这样的问题：在实际中经常出现一些数值较小的 N_r 和 $n_{r,i}$（$N_r=1$，$n_{r,i}=0$）。因此，通常会在 $P(k_i|R)$ 和 $P(k_i|\overline{R})$ 的估计式中添加一个 0.5 的调节系数，这样可以得到：

$$P(k_i|R)=\frac{n_{r,i}+0.5}{N_r+1}$$

$$P(k_i|\overline{R})=\frac{n_i-n_{r,i}+0.5}{N-N_r+1} \tag{5-7}$$

这种相关反馈过程的主要优点是反馈过程是和查询项新权重的生成直接相关。缺点包括：1）在反馈循环中没有考虑文档索引项权重；2）忽略了之前的查询表式中项的权重；3）没有使用查询扩展（原始查询中相同一组项被一次次重新确定权重）。由于这些缺点，概率相关反馈方法总体上不如传统的向量修改模型有效。

为了扩展概率模型的查询扩展能力，很多文献中提出了许多不同的方法，包括查询扩展中的权重计算，以及基于生成树的索引项聚类方法等。所有这些方法都把概率查询扩展区别于概率查询项再赋权。尽管我们在这里不讨论，但在 5.8 节中介绍了对这个问题进行研究的简史。

5.3.3 相关反馈的评价

考虑由 Rocchio 式生成了改进的查询向量 $\vec{q}_m$，假设我们需要评估它的检索质量。一个简单的方法是用 $\vec{q}_m$ 检出一个文档集，用向量公式对它们排序，由专家来度量相对于原始查询向量 $\vec{q}$ 的相关文档集的召回率-精度值。通常，这个结果显示了巨大的提升。不幸的是，其中很大一部分提升是源自在反馈的过程中已经标为相关文档的排名被进一步提升了[582]。由于用户已经见过这些文档（并指出它们是相关的），因此这样的评价是不真实的。而且，由于有些文档用户没有见到，这就掩盖了检索质量的真实提高。

一个更实际的方法是仅在剩余文档集（residual collection）上评价改进的查询向量 $\vec{q}_m$，即所有文档集减去由用户提供的反馈文档集。因为那些排名靠前的文档已经从文档集中删去了，所以 $\vec{q}_m$ 的召回率-精度要比原始的查询向量 $\vec{q}$ 低。这不是一个缺点，因为主要目的是比较不同的相关反馈策略的质量（而不是比较反馈前后的质量）。因此，一个基本的经验法则
184 是，任何包含了相关反馈策略的实验应该总是计算相对于剩余文档集的召回率-精度。

5.4 基于点击的显式反馈

Web 搜索引擎用户不仅观察查询的答案，而且还要点击它们。这些点击，反映了他们对于给定查询特定答案的偏好，可以在不干扰用户的情况下大量收集。人们自然会问，它们是否也反映了答案的相关性评价，即点击数据是否能用来判断未来查询的相关性。在一定的条件下，答案是肯定的。本节基于文献［845，1320］讨论这个问题。

5.4.1 眼动追踪和相关性评价

点击数据提供了有限的用户行为信息。一种补充用户行为信息的方法是用眼动追踪（eye tracking）设备。

1. 眼动追踪

使用商业设备可以追踪用户目光的位置，这些设备采用了基于瞳孔中心和角膜反射等方

法来确定用户聚焦在屏幕的哪个区域。这个方法尽管在一些实例中有错误，但仍可以以60％～90％的正确率探测到用户在屏幕中关注的区域，并且可以确定何时这个方法无效。

遵照 Rayner[1339]，眼球运动可以分为四个基本行为类型：凝视（fixation）、扫视（saccade）、瞳孔扩张（pupil dilation）和扫视顺序（scan path）。凝视是盯着屏幕特定区域持续 200～300 毫秒，这段时间长到足以让大脑有效地捕获和翻译显示的图像。也就是说，凝视通常代表与视觉信息获取和处理相关的视觉活动，对于解释用户行为是很重要的。扫视是在凝视点间持续 40～50 毫秒的快速目光运动，因为太短而不能有效获取信息。瞳孔扩张，是表示感兴趣，与解释用户行为有关，这里不讨论。这里也不讨论扫视顺序。

本节后面报告的实验结果是［845，1320］实验的重现，完全基于凝视。尽管有所限制，但它们确实提供了足够的信息来说明用户对一组相对于用户查询结果的解释。

2. 相关性评价

为了评价结果的质量，眼动追踪是不合适的。正如在第 4 章中讨论的，结果的质量评价需要选择一组测试查询，确定它们的相关性评价。我们如果要评价点击产生的质量也是如此。

举例来说，［845］中的研究使用了一组由 10 个查询组成的测试集，其中 5 个是浏览型的，即找到一个网页；5 个是信息型的，即找到关于一个主题的信息。其中关于浏览型的例子是“find the homepage of Michael Jordan，the statistician”（找到统计学家 Michael Jor- 185
dan 的主页）。信息型任务的例子是“find the location of the tallest mountain in New York”（找到纽约最高的山的位置）。为了编辑点击数据，需要雇用真实的用户。可以使用一个代理来监视并管理用户和搜索引擎间的互动，这样可以不影响用户的体验。

用户浏览的每一个页面和结果集由独立的评估员评价相关性。对于给定查询以及与这个查询相关的前 10 个结果，评估员需要对这 10 个结果确定一个严格的排序。设 r_i 表示第 i 个结果。如果检索结果的偏序中 r_4 在 r_2 的前面出现，评估员就认为 r_4 比 r_2 更可能产生和查询相关的信息。为了减少噪声和错误评价的影响，每个查询应该由至少两个评估员评价。当这两个评估员对一对结果的相关性偏好一致时，就产生了一个强的相关性评价得分。

由评估员产生的相关性评价可以用来评价用户点击的文档和测试查询集中每个查询的相关性。

5.4.2　用户行为

在［845］报告的眼动追踪实验中，包含了 29 个测试者。实验表明用户自顶向下扫视查询结果，他们在第二次或者第三次凝视中，检查排名第一和排名第二的结果。并且，他们趋于彻底地扫视前 5 个或者前 6 个出现在屏幕可视区域的答案，之后滚动检查前 10 个答案中剩余的部分。

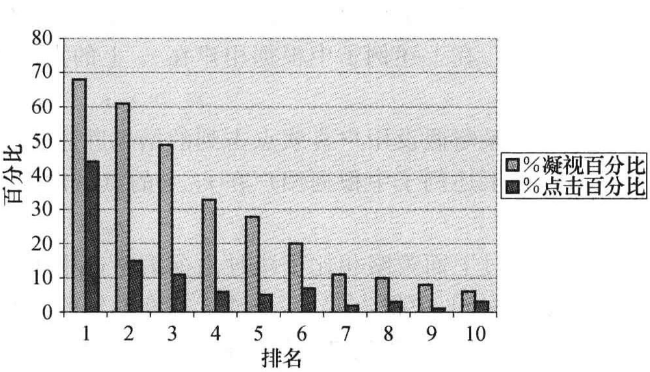


图 5-4　用户浏览和点击每个排名靠前的结果的比例。该结果是从［845］中重现的

图 5-4 显示了在［845］中 10 项测试任务和 29 个测试者的凝视和点击的百分比。我们注意到在 60％～70％的任务中，用户凝视第一个或者第二个结果。对第四个结果的凝视下

186 降到了一半。对于页面的底部，只能通过滚动才能检查，凝视的相对频度低于10%。结果也表明用户几乎平等地观察前两个答案，但是他们点击第一个结果的频率几乎是第二个的3倍。这可能表明了用户对搜索引擎的偏好，即用户趋于信赖搜索引擎推荐的第一个结果。

这个现象还可以通过额外的实验来更好地解释，其中对测试者给定两个不同的结果集：1）由搜索引擎返回的正常排序；2）一个修改过的排序，其中前两个结果的位置对换了。通过代理使得这些修改对用户是不可见的，这样测试者不知道他看到的是正常的还是修改过的排序。在[845]中的结果分析表明用户在搜索引擎中表现出信赖偏好（trust bias），喜欢第一个结果，即使当前两个结果被调换了，用户点击第一个位置的频率相比于第二个位置多出3倍。也就是说，结果的位置对用户决定是否点击它们有很大的影响。

5.4.3 点击作为用户偏好的指标

通过上述的讨论，我们可以看到把点击解释为相关性的直接指示不是最好的方法。更好的是把点击解释为用户偏好的指标。举例来说，如果用户查看了结果的片断，即在每一个搜索引擎下面显示的文本短摘要之后，用户决定要跳过它并点击在排序中低于它的结果，就可以说相比于排序中上面的结果，这个用户更喜欢这个被点击的结果。这种类型的偏好关系不仅要考虑用户的点击结果，也要考虑观察到、但没有被点击的结果。

1. 在相同查询内的点击

为了把点击解释为用户的偏好，我们采用如下的定义。

定义 *给定一个排序函数$\mathcal{R}(q_i, d_j)$，设r_k是第k个排序结果。也就是说，r_1、r_2、r_3分别表示第一、第二和第三个结果。并且，设“$\surd r_k$”表示用户点击了第k个结果。定义偏好函数“$r_k > r_{k-n}$”，$0 < k-n < k$，这表示根据用户的点击行为，相比于第（$k-n$）篇文档，用户更倾向于第k篇文档。*

举例来说，假设如下的关于用户对一组查询结果的点击行为的例子：

$$r_1 \quad r_2 \quad \surd r_3 \quad r_4 \quad \surd r_5 \quad r_6 \quad r_7 \quad r_8 \quad r_9 \quad \surd r_{10}$$

用户点击了r_3、r_5和r_{10}的结果，但这不能让我们做出它们和查询之间明确的相关性。但是，它让我们得出这个用户相对的偏好。为了捕获这个例子中的偏好关系，在[845]中提出了两种不同的方法，如下所示。

- 跳过上面（Skip-Above）：如果$\surd r_k$，那么对于所有没有点击到的r_{k-n}有$r_k > r_{k-n}$。
187 - 跳过之前（Skip-Previous）：如果$\surd r_k$，并且r_{k-1}没有点击到，那么$r_k > r_{k-1}$。

第一个策略假定用户喜欢点击到的结果胜过所有没有点击到但排在前面的其他结果。根据这种策略，在上述例子中根据用户在r_3上的点击产生如下的偏好：

$$r_3 > r_2; \qquad r_3 > r_1$$

第二个策略假设用户喜欢点击到的结果胜过排在它前一位并且没有点到的结果。根据这个策略，在上述例子中根据用户在r_3上的点击产生如下的偏好：

$$r_3 > r_2$$

注意跳过上面策略相比于跳过之前策略产生了更多的偏好关系。

根据[845]中报告的10个测试查询的经验结果，可以获得如下的结论：

- 跳过上面和跳过之前策略都能产生大约80%与评估员相一致的相关性偏好关系。
- 如果我们在呈现给用户前交换第一个和第二个结果，那么它们的点击仍然反映了与大约80%的评估员的相关性评价相吻合的偏好关系。

- 如果我们在呈现给用户前颠倒前 10 篇结果的顺序，那么对于这两种策略，它们的点击仍然反映了与大约 80%的评估员的相关性评价相吻合的偏好关系。

这些结果提供了坚实的证据，表明用户的点击不仅可用来表示个性化偏好，而且可用来表示给定查询结果的相对相关性。

2. 查询链的点击

上述的讨论限制在单一查询的情况下。然而，在实际应用中，用户对同一个任务搜索答案时会提出不止一个查询。与同一个任务对应的查询集可以看做是实时的查询流，并构成所谓的*查询链*。在查询链中分析点击的目的是产生新的偏好关系，从而对那些在同一个查询中产生的偏好关系进行补充。

举例来说，假设同一个查询链的两个结果可以导致如下的点击行为：

$$r_1 \quad r_2 \quad r_3 \quad r_4 \quad r_5 \quad r_6 \quad r_7 \quad r_8 \quad r_9 \quad r_{10}$$

$$s_1 \quad \surd s_2 \quad s_3 \quad s_4 \quad \surd s_5 \quad s_6 \quad s_7 \quad s_8 \quad s_9 \quad s_{10}$$ 188

其中 r_j 表示在第一个结果集中的答案，s_j 表示第二个结果集中的答案，$\surd s_k$ 表示第二个结果集中第 k 个答案被点击了。在这种情况下，用户没有点击第一个结果集中的答案，而是点击了第二个结果集的第二个和第五个答案。

在［845］中，推荐了两个不同的捕获偏好关系的策略，如下所示。

- Top-One-No-Click-Earlier：如果 $\exists s_k \mid \surd s_k$，那么对所有的 $j \leqslant 10$，有 $s_j > r_1$。
- Top-Two-No-Click-Earlier：如果 $\exists s_k \mid \surd s_k$，那么对所有的 $j \leqslant 10$，有 $s_j > r_1$ 且 $s_j > r_2$。

第一个策略假设用户跳过第一个结果集的所有 10 个答案，而去点击第二个结果集，说明这个用户喜欢第二个集合中任何一个答案胜过第一个结果集中的第一个答案。根据这样的策略，由用户在上面例子的结果 s_2 上的点击产生了如下的偏好关系：

$$s_1 > r_1; \quad s_2 > r_1; \quad s_3 > r_1; \quad s_4 > r_1; \quad s_5 > r_1; \quad \cdots$$

第二个策略假设用户跳过了第一个结果集的所有 10 个答案，而去点击第二个结果集，说明用户喜欢第二个集合中任何一个答案胜过第一个结果集中的前两个答案。根据这样的策略，由用户在上面例子的结果 s_2 上的点击产生了如下的偏好关系：

$$s_1 > r_1; \quad s_2 > r_1; \quad s_3 > r_1; \quad s_4 > r_1; \quad s_5 > r_1; \quad \cdots$$

$$s_1 > r_2; \quad s_2 > r_2; \quad s_3 > r_2; \quad s_4 > r_2; \quad s_5 > r_2; \quad \cdots$$

我们注意到，Top-Two-No-Click-Earlier 策略产生了两倍于 Top-One-No-Click-Earlier 策略的偏好关系。

为了验证这些偏好关系的有效性，我们把它们和评估员的相关性评价结果进行比较。根据［845］中报告的结果，对于一组 10 个测试查询，可以获得如下的结论：

- Top-One-No-Click-Earlier 和 Top-Two-No-Click-Earlier 策略产生了与大约 80%的评估员的相关性结果相一致的偏好关系。
- 即使在呈现给用户前，我们调换第一个和第二个结果，仍然可以观察到类似的一致性。
- 即使在呈现给用户前，我们颠倒结果的排序，仍然可以观察到类似的一致性。

这些结果表明用户不仅（通过不点击）提供了对整个结果的负面反馈信息，也从这个过程中学习，并在接下来的步骤中重构更好的查询。

考虑到用户的点击经常反映出与评估员的相关性评价结果相一致的偏好关系，人们可以考虑使用点击信息来提高排序，正如我们将在 11.5.4 节排序学习中讨论的那样。 189

5.5 通过局部分析的隐式反馈

在局部反馈策略中，在查询阶段，根据给定查询 q 所检出的文档来决定用于查询扩展的索引项，如图 5-2a 所示。这类似于相关反馈循环，但却是在没有用户协助的情况下完成的。这里讨论两个局部策略：局部聚类和局部上下文分析。前者是基于 Attar 和 Fraenkel[78] 的早期工作，用于建立查询扩展应用聚类技术的许多基础性想法和概念。后者是 Xu 和 Croft[1732] 的工作，显示了同时结合局部和全局分析的优点。

5.5.1 通过局部聚类的隐式反馈

查询扩展中使用的聚类技术从早期开始就是信息检索中的一项基本方法。其标准的过程是建立类似于关联矩阵那样的全局结构，量化索引项之间的相互关系，然后把相互关联的索引项用于查询扩展中。主要的问题是对于一般文档集，全局结构并不总是能有效地提高检索质量。一个主要的原因是全局结构可能不能很好地适应由当前查询定义的局部上下文。为了解决这个问题，可以使用局部聚类方法。我们的讨论基于 [78]。

定义 *对于给定的查询 q，检出文档集 D_l 称为局部文档集。设 N_l 是 D_l 中文档的数量。而且，在局部文档集中所有不同的索引项组成的集合 V_l 称为局部词汇表。索引项 k_i 在文档 d_j，$d_j \in D_l$ 中出现的频度是 $f_{i,j}$。设 $\boldsymbol{M}_l=[m_{i,j}]$ 是一个项-文档矩阵，V_l 行 N_l 列，其中 $m_{i,j}=f_{i,j}$。考虑到 $\boldsymbol{M}_l^{\mathrm{T}}$ 是 $\boldsymbol{M}_l$ 的转置，矩阵 $\boldsymbol{C}_l=\boldsymbol{M}_l\boldsymbol{M}_l^{\mathrm{T}}$ 是局部项间相关性矩阵。每个元素 $c_{u,v} \in \boldsymbol{C}_l$ 表达了索引项 k_u 和 k_v 之间的关系。*

局部项间相关性矩阵 $\boldsymbol{C}_l$，类似于在第 3 章中在整个文档集上定义的项间相关性矩阵。这里它基于查询 q 返回的文档内索引项的共现，来建立两个项 k_u 和 k_v 之间的关系。这两个项共现的文档数量越高，相关性就越强。

为了利用类似于项间距离这种因素的优点，相关性强度可以用不同的方式定义。当已经计算了所有的相关性强度时，它们可用于计算邻近索引项组成的局部簇。在同一个簇中的索引项可以用来做查询扩展。我们这里考虑三种簇：关联簇（association cluster）、度量簇（metric cluster）和标量簇（scalar cluster）。

1. 关联簇

定义 *一个关联簇是通过重新定义任意索引项对 k_u 和 k_v 之间的相关因子 $c_{u,v}$，而从局*
190 *部相关性矩阵 $\boldsymbol{C}_l$ 中计算的，如下所示：*

$$c_{u,v}=\sum_{d_j \in D_l} f_{u,j} \times f_{v,j} \tag{5-8}$$

在这种情况下，相关性矩阵（correlation matrix）称为局部关联矩阵（local association matrix）。

其潜在的动机是在文档中经常共现的索引项有同义关系。而且，式（5-8）简单，容易掌握，且相当直接。相关因子 $c_{u,v}$ 量化了共现的绝对频度，可以称为是非归一化的。因此，相关性矩阵 $\boldsymbol{C}_l$ 也称为是非归一化的。另外一种方法是对相关性矩阵进行归一化。举例来说，

$$c'_{u,v}=\frac{c_{u,v}}{c_{u,u}+c_{v,v}-c_{u,v}} \tag{5-9}$$

在这个例子中，相关性矩阵 $\boldsymbol{C}_l$ 称为是归一化的。

给定一个局部相关性矩阵 $\boldsymbol{C}_l$，我们能利用它按如下方式构建局部关联簇。

定义 *设 $C_u(n)$ 是一个从局部相关性矩阵 $\boldsymbol{C}_l$ 中返回最大的 n 个相关因子 $c_{u,v}$ 的函数，*

其中 v 遍历局部索引项集合，且 $v \neq u$。这样，$C_u(n)$ 定义了局部关联簇，即围绕索引项 k_u 的一个邻域。如果 $c_{u,v}$ 是由式（5-8）给定的，关联簇称为是非归一化的。如果 $c_{u,v}$ 是由式（5-9）给定，则关联簇称为是归一化的。

给定查询 q，我们通常只感兴趣于找到 $|q|$ 个查询项的关联簇。而且，希望能保持关联簇的规模小。这意味着，如果能维护合适的索引结构，那么在提交查询时能有效地计算这样的关联簇。

2. 度量簇

关联簇是基于文档中索引项的共现，不考虑项出现在文档中的位置。由于出现在同一句子中的两个索引项比散落在文档中的两个项联系得更紧密，因此可能值得把两个索引项的距离因素考虑在相互关系的计算中。度量簇就是基于这样的想法。

定义 度量簇是从局部相关矩阵 $\boldsymbol{C}_l$ 中计算得来，把任意一对索引项 k_u 和 k_v 之间的相关因子 $c_{u,v}$ 重新定义为它们在文档中距离的函数。假设 $k_u(n, j)$ 是一个返回了索引项 k_u 在文档 d_j 中的第 n 次出现的函数。而且，假设 $r(k_u(n, j), k_v(m, j))$ 是表示在文档 d_j 中索引项 k_u 的第 n 次出现和索引项 k_v 的第 m 次出现的距离的函数。这个距离可以通过两个共现的索引项间词的个数来计算。定义：

$$c_{u,v} = \sum_{d_j \in D_l} \sum_n \sum_m \frac{1}{r(k_u(n,j), k_v(m,j))} \tag{5-10}$$

在这个例子中，相关性矩阵称为局部度量矩阵。 191

注意如果 k_u 和 k_v 出现在不同的文档中，我们把它们的距离定为无穷大。已有文献给出了上述 $c_{u,v}$ 表达式的变体，例如 $1/r^2(k_u(n, j), k_v(m, j))$。

度量相关因子 $c_{u,v}$ 量化了绝对反比距离，称为是非归一化的。这样，局部度量矩阵 $\boldsymbol{C}_l$ 称为是非归一化的。另外一种方法是归一化相关因子。例如，

$$c'_{u,v} = \frac{c_{u,v}}{[k_u, k_v]\text{ 对的总数}} \tag{5-11}$$

在这个例子中，局部度量矩阵 $\boldsymbol{C}_l$ 称为是归一化的。

给定一个局部度量矩阵 $\boldsymbol{C}_l$，我们能利用它按照如下的方法建立局部度量簇。

定义 假设 $C_u(n)$ 是一个函数，返回局部度量矩阵 $\boldsymbol{C}_l$ 中前 n 个大的 $c_{u,v}$ 值，$v \neq u$。那么，$C_u(n)$ 定义了围绕着索引项 k_u 的一个局部度量簇。如果 $c_{u,v}$ 是由式（5-10）定义的，那么度量簇称为是非归一化的。如果 $c_{u,v}$ 是由式（5-11）定义的，那么度量簇称为是归一化的。

3. 标量簇

另一种导出两个局部索引项 k_u 和 k_v 同义关系的方法是比较集合 $C_u(n)$ 和 $C_v(n)$，即比较两个项的邻域。其想法是两个有着类似邻域（neighborhood）的项有着同义关系。在这种情况下我们说这种关系是间接的，或是由邻域导出的。一种量化邻域关系的方法是把索引项 k_u 中所有的相关因子 $c_{u,x}$ 转化为一个向量 $\vec{s}_u$，索引项 k_v 中所有的相关因子 $c_{v,y}$ 转化为另一个向量 $\vec{s}_v$，通过标量比较这两个向量。一种常见的标量相似度测度是两个向量之间的夹角余弦。

定义 假设 $\vec{s}_u = (c_{u,x_1}, c_{u,x_2}, \cdots, c_{u,x_n})$，$\vec{s}_v = (c_{v,y_1}, c_{v,y_2}, \cdots, c_{v,y_m})$ 分别是 k_u 和 k_v 的邻域相关因子组成的向量。定义：

$$c_{u,v} = \frac{\vec{s}_u \cdot \vec{s}_v}{|\vec{s}_u| \times |\vec{s}_v|} \tag{5-12}$$

在这个例子中，相关性矩阵 $\boldsymbol{C}_l$ 是局部标量矩阵。

局部标量矩阵 $\boldsymbol{C}_l$ 称为从邻域中导出的。使用它，可以定义如下的标量簇。

定义 假设 $C_u(n)$ 是一个函数，返回局部标量矩阵 $\boldsymbol{C}_l$ 中前 n 个大的 $c_{u,v}$ 值，$v\neq u$，$c_{u,v}$ 的值是根据式（5-12）定义的。那么，$C_u(n)$ 定义了围绕着索引项 k_u 的一个标量簇。

4. 基于邻居项的查询扩展

与查询项关联的簇中的索引项可以用于扩展原始查询。这样的项称为查询项的邻居，可
192 以按如下方式描述。

属于簇 $C_u(n)$ 的索引项 k_v 和索引项 k_u 关联，这个项称为是 k_u 的邻居。有时候 k_v 也称为 k_u 的搜索同义项（searchonym），这里我们使用术语*邻居*（neighbor）。尽管邻居项被认为有同义关系，但它们不一定在语法上有同义关系。通常，邻居项代表了当前查询背景下相互关联的不同关键词。相关性矩阵中的文档和索引项的局部性反映了这种相互关系的局部性。从广义上说，由于它们能用来扩展搜索表式，因此，尽管在查询表式中没有明确表达，但它们仍朝着用户希望的方向发展，说明了邻居项是局部聚类过程的重要产出。

考虑用邻居项来扩展一个给定的用户查询 q。一种可能性是按照如下的方式扩展查询。对于每个项 $k_u\in q$，从簇 $C_u(n)$（可以是关联、度量，或者标量簇）中选择 m 个邻居项，把它们添加到查询中。可以用如下的方式表达：

$$q_m = q \cup \{k_v \mid k_v \in C_u(n), k_u \in q\}$$

希望其他的邻居项 k_v 可以检索出新的相关文档。为了涵盖一个更广阔的邻域，集合 $C_u(n)$ 可能是使用归一化和非归一化的相关因子获得的查询项。定性的解释是非归一化簇倾向于把出现频率高的项聚在一起，而归一化簇倾向于把更稀少的项聚在一起。这样，两个簇的并集提供了一个可能更好的相互关系表示。

查询扩展是重要的，因为能检出更多的文档，有可能提高召回率。然而，要排序的文档更多也意味着精度更低，即新检出的不相关文档会有高的排序。因此，需要小心使用查询扩展，并根据手头的文档集进行精心的调整。

5.5.2 通过局部上下文分析的隐式反馈

上面讨论的局部聚类技术是基于原始查询的检出文档集，从排序靠前的文档中聚合出邻居项。另外一种方法是在整个文档集中搜索查询项的相关性——称为全局分析的方法(global analysis)。全局技术通常是构建一个同义词典来包含整个文档集中索引项的相互关系。这些项看做是概念，同义词典看做是概念关系结构。

同义词典构建代价昂贵，除了提供对查询扩展的支持外，作为浏览工具也是有用的。同义词典的构建通常考虑使用小的上下文和短语结构，而不是简单地采用由整篇文档提供的上下文。而且，使用全局分析的现代变体，选择整个查询（而不是单个查询项）最接近的项用于查询扩展。本节的主题是把全局分析的想法（例如小的上下文和短语结构）用于检出的局
193 部文档。

局部上下文分析 [1732]，是一种结合了全局分析和局部分析的方法，基于名词组的使用，即单一的名词、相邻的两个名词，或者文本中相邻的三个名词，而不是简单的关键词。从排名靠前的文档中选出的名词组是文档中可用于查询扩展的概念。该方法使用段落，即一个固定的文本窗口，而不是文档来决定索引项的共现。

更具体地说，局部上下文分析过程包含了三步。

- 第一步，使用原始查询检出前 n 个段落。这是通过把查询初始检出的文档分为固定长度的段落（比如，300个词），把这些段落按文档的方式排序。

- 第二步，对于每一个排名靠前的段落中的概念 c(即，名词组)，使用 TF-IDF 排序的变体计算整个查询 q(而不是单个的查询项) 和概念 c 的相似度 $sim(q, c)$。
- 第三步，根据 $sim(q, c)$ 排序的前 m 个概念被加入到原始查询 q 中。对于每个添加的概念，赋予一个权重 $1-0.9\times i/m$，其中 i 是概念的排名。在原始查询中的项可以通过对每个项赋予权重 2 来加强。

这三步中，第二步是最复杂的，我们现在来讨论一下。每个概念 c 和原始查询 q 之间的相似度 $sim(q, c)$ 按照如下方式计算。

$$sim(q,c) = \prod_{k_i \in q} \left(\delta + \frac{\log(f(c,k_i) \times IDF_c)}{\log n}\right)^{IDF_i} \tag{5-13}$$

其中 n 是要考察的排名靠前的段落的数量。函数 $f(c, k_i)$ 量化了概念 c 和查询项 k_i 的相关性，如下所示。

$$f(c,k_i) = \sum_{j=1}^{n} pf_{i,j} \times pf_{c,j}$$

其中 $pf_{i,j}$ 是查询项 k_i 在第 j 个段落中的频度，$pf_{c,j}$ 是概念 c 在第 j 个段落中的频度。注意这是为关联簇定义的标准相关因子，参见式 (5-8)，但是迁移到了段落上。反比文档频率系数可以计算为：

$$IDF_i = \max\left(1, \frac{\log_{10}(N/np_i)}{5}\right)$$

$$IDF_c = \max\left(1, \frac{\log_{10}(N/np_c)}{5}\right)$$

其中 N 是段落的数量，np_i 是包含查询项 k_i 的段落数量，np_c 是包含了概念 c 的段落数量。式 (5-13) 中的系数 δ 是一个常数，用来避免在乘积运算中引入值为零的系数。通常 δ 的 194 值接近 0.1(最大值 1 的 10%)。最后，在指数中的 IDF_i 的参数是用来加强查询项的频度的。

上述计算 $sim(q, c)$ 的过程是一个 TF-IDF 排序公式的变体。并且，它已经为 TREC 数据做了调整，在其他文档集上不是那么有效。因此，有一点很重要，需要记住，即不同的文档集可能需要调整参数。我们也注意到在局部上下文分析中使用到的相关性测度是关联型的。然而，我们已经知道度量型的相关性测度是更有效的。因此，剩下来的就是为函数 $f(c, k_i)$ 测试度量型相关因子是否能在局部上下文分析中有所区别。

5.6 通过全局分析的隐式反馈

上面讨论的局部分析方法从局部检出的文档中抽取信息来扩展查询。大家普遍认为，对于不同的文档集，一旦精细地调整过，那么局部分析就能产生更好的检索质量。另一种方法是使用从整个文档集中的信息来扩展查询。基于这个想法的策略称为全局分析过程。

接下来，我们讨论两个全局分析的现代变体。两者都是基于类似于同义词典结构建立起来的，使用了文档集中的所有文档。然而，用来构建同义词典的方法和为查询扩展选择索引项的过程在这两种情况中都是不同的。

5.6.1 基于相似度同义词典的查询扩展

本节中，我们讨论一种基于自动构建的全局相似度同义词典的查询扩展方法 [1309]。相似度同义词典是基于索引项间的关系，而不是共现矩阵。在下面的讨论中将区分清楚。此外，要特别关注用于扩展的索引项的选择，和对这些项权重的再赋权。与之前的全局分析方

法不同，用于扩展的索引项是基于它们和整个查询的相似度进行挑选的，而不是基于它们和单个查询项的相似度。

使用项间关系可以构建一个相似度同义词典，这是源自于把所有的项看做是概念空间中的概念。在这个概念空间中，每个项用它出现的文档索引。于是，项承担了文档原先的作用，而文档解释为索引单元。接下来的定义建立了一个合适的框架。

定义 如前所述（见第 3 章），假设 t 是文档集中项的个数，N 是文档集中文档的个数，$f_{i,j}$ 是项 k_i 在文档 d_j 中出现的次数。而且，设 t_j 是文档 d_j 中不同项的个数，ITF_j 是文档 d_j 的反比项频，定义：

$$ITF_j = \log\left(\frac{t}{t_j}\right)$$

195 这是类似于反比文档频率的定义。

在这个框架内，每个项 k_i 被赋予一个向量 $\vec{k}_i$，如下所示。

$$\vec{k}_i = (w_{i,1}, w_{i,2}, \cdots, w_{i,N})$$

其中，正如第 3 章中那样，$w_{i,j}$ 是项-文档对 $(k_i,\ d_j)$ 的权重。然而在这里，文档权重用来表示项向量，而不是像 3.2.6 节中的经典向量模型那样用项权重来表示文档向量。并且，这些权重是用不同方式计算的，如下所示。

$$w_{i,j} = \frac{\left(0.5 + 0.5\,\frac{f_{i,j}}{max_j(f_{i,j})}\right) ITF_j}{\sqrt{\sum_{l=1}^{N}\left(0.5 + 0.5\,\frac{f_{i,l}}{max_k(f_{i,k})}\right)^2 ITF_l^2}} \tag{5-14}$$

其中 $max_j(f_{i,j})$ 计算第 i 项的所有 $f_{i,j}$ 系数的最大值，即遍历了文档集的所有文档。我们注意到上面的表达式是 TF-IDF 权重的变体，但是采用了反比项频作为代替。

两个项 k_u 和 k_v 之间的关系计算为相关因子 $c_{u,v}$，如下所示。

$$c_{u,v} = \vec{k}_u \cdot \vec{k}_v = \sum_{\forall d_j} w_{u,j} \times w_{v,j} \tag{5-15}$$

我们注意到相关性测度是用于计算标量矩阵的相关因子的变体，由式（5-12）定义。主要的区别是，权重是基于把文档解释为索引单元，而不是把文档看做项共现的场所。全局相似度同义词典是由式（5-15）定义的相关因子 $c_{u,v}$ 组成的标量项间矩阵。当然，计算这个矩阵是代价昂贵的。但是，这个全局相似度同义词典只需要计算一次，并且可以被增量式地更新。

给定全局相似度同义词典，查询扩展是按照如下三步进行的。

- 第一步，在用于表示索引项的同一个向量空间中表达查询。
- 第二步，根据全局相似度同义词典，计算每个与查询项相关的索引项 k_v 和整个查询 q 的相似度 $sim(q,\ k_v)$。
- 第三步，根据 $sim(q,\ k_v)$，用前 r 个索引项来扩展查询。

对于第一步，查询表达为如下的向量空间中的形式。

定义 查询 q 被赋予一个向量 $\vec{q}$，如下所示：

$$\vec{q} = \sum_{k_i \in q} w_{i,q}\, \vec{k}_i$$

其中 $w_{i,q}$ 是和项-查询对 $[k_i,\ q]$ 关联的权重。这个权重是由式（5-14）给定的，其中文档
196 d_j 被查询 q 替换了。

对于第二步，每个与查询项相关的索引项 k_v 和用户查询 q 的相似度 $sim(q,\ k_v)$ 计算为：

$$sim(q,k_v) = \vec{q} \cdot \vec{k}_v = \sum_{k_j \in q} w_{i,q} \times c_{i,v} \quad (5\text{-}16)$$

其中，$c_{i,v}$是由式（5-15）给定的相关因子。如图 5-5 所示，相对于单个的查询项，索引项 k_v 可能更接近整个查询的中心 q_c。这意味着，这里选出的用于查询扩展的索引项可能和之前全局分析方法选出的项不同，之前是用相对于单个查询项的相似度函数来决定查询扩展的索引项，这样会选择 k_i 而不是 k_v 来扩展查询。

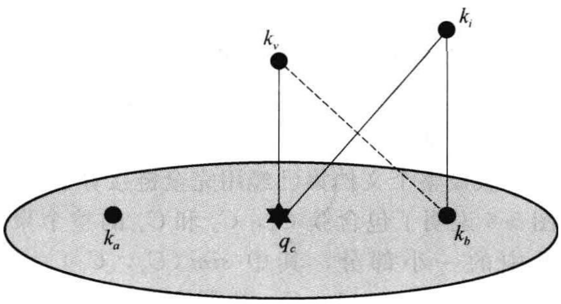


图 5-5　索引项 k_v 到查询中心 q_c 的距离可能与 k_v 到单个查询项的距离完全不同

对于第三步，把根据相似度 $sim(q, k_v)$ 排序的前 r 个索引项加入到原始查询 q 形成扩展查询 q_m。对于查询 q_m 中的每个扩展项 k_v 赋予一个权重 w_{v,q_m}，如下所示。

$$w_{v,q_m} = \frac{sim(q,k_v)}{\sum_{k_i \in q} w_{i,q}}$$

扩展查询 q_m 用来检索出新的文档。与之前的全局分析方法不同，这个技术在三个不同的文档集中都取得了更好的检索质量（20%的幅度）。

值得注意的是，假设文档 d_j 在项向量空间中表示为$\vec{d}_j = \sum_{k_i \in d_j} w_{i,j} \vec{k}_i$。而且，假设原来的查询 q 扩展后包括了文档集中所有的（赋予适当权重的）的 t 个索引项。这样，文档 d_j 和查询 q 之间的相似度 $sim(q, d_j)$ 可以在项向量空间内计算为：

$$sim(q,d_j) = \sum_{k_v \in d_j} \sum_{k_u \in q} w_{v,j} \times w_{u,q} \times c_{u,v} \quad (5\text{-}17)$$ 197

这样的表达式类似于广义向量空间模型（见 3.4.1 节）中的查询-文档相似度。这样，广义向量空间模型可以解释为查询扩展技术。其主要的区别是权重计算，以及在这里描述的索引项-概念技术只使用前 r 个项来扩展。

5.6.2　基于统计同义词典的查询扩展

本节中，我们讨论基于全局统计同义词典 [457] 的查询扩展技术。这与上面所描述的基于相似度同义词典的方法有很大不同。

全局同义词典是由整个文档集中相互关联的索引项组成的同义词类构成的。这些相互关联的索引项可以用来扩展原来的用户查询。为了取得更好的效果，选出的扩展项必须有较高的区分度 [1417]，这意味着它们必须是低频项。然而，由于低频项的信息量少（它们出现在少数几篇文档中），因此难于对它们进行有效的聚类。为了避免这个问题，把文档聚成簇，并用簇内文档中低频项的集合来定义同义词类。这确保了文档聚类算法产生小而紧的簇。

一个产生小而紧的簇的文档聚类算法是*完全链接算法*（complete link algorithm），它的流程如下（基本表示）：

1）首先，把每个文档放在一个不同的簇中。

2）计算所有簇间的相似度。

3）找到具有最高簇间相似度的 [C_u，C_v]。

4）合并簇 C_u 和 C_v。

5）验证停止条件。如果这个条件不满足，回到第 2 步。

6）返回层次化的聚类结果。

两个簇间的相似度定义为所有簇间文档对（即不在同一个簇中的两篇文档）的最小相似度。为了计算一对文档之间的相似度，使用向量模型中的余弦公式。作为这个最小准则的结果，得到的簇趋于小而紧。

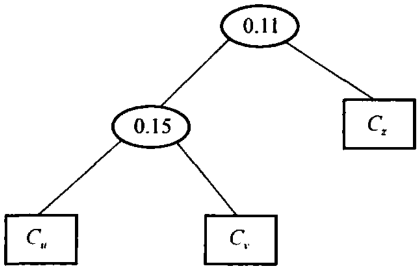


图 5-6 由完全链接算法生成的三个簇的层次体系。簇间相似度在椭圆中标出

假设整个文档集已经用完全链接算法进行了聚类。图 5-6 说明了包含簇 C_u，C_v 和 C_z 的整个聚类层次体系中的一小部分，其中 $sim(C_u, C_v)=0.15$，$sim(C_{u+v}, C_z)=0.11$，C_{u+v}表示合并 C_u 和 C_v 得到的簇。我们注意到当层次体系上升时，相似度下降，这是因为高层的簇包括了更多的文档，表示更松散的组合。因此，最紧密的簇位于聚类体系的底部。

给定整个文档集的聚类层次结构，全局同义词典中组成每个同义词类的索引项是按照如
198 下的方式挑选的。

- 从用户获得三个参数：阈值类（Threshold Class，TC）、簇中的文档数（Number of Documents in a Class，NDC）和最大反比文档频率（Maximum Inverse Document Frequency，MIDF）。
- 使用参数 TC 作为判定用于生成同义词类的文档簇的阈值。如果簇 C_u 和 C_v 中的文档被选为某个索引项同义词类的来源，那么这个阈值必须高于 $sim(C_u, C_v)$。例如，在图 5-6 中，如 TC 的值为 0.14，则返回一个同义词类 C_{u+v}，而 0.10 的 TC 值可以返回类 C_{u+v}和 C_{u+v+z}。
- 使用参数 NDC 作为要考察的簇的数量限制（文档数量）。例如，如果 C_{u+v}和 C_{u+v+z}是（通过参数 TC）预选的，那么参数 NDC 可能用来在这两者间做决定。较低的 NDC 值可能把挑选限制在更小的簇 C_{u+v}中。
- 考察如上（通过参数 TC 和 NDC）预选的每个文档簇。只有低频项被用作索引项同义词类的来源集合。参数 MIDF 定义了被选中加入同义词典的索引项反比文档频率的最小值。这样做才可能确保仅有低频项加入到生成的同义词典中（太一般的项不是好的同义词）。

给定已经构建好的同义词类，它们能用于查询扩展。对此，每个同义词类 C 的平均项权重 wt_C 计算如下：

$$wt_C = \frac{\sum_{i=1}^{|C|} w_{i,C}}{|C|}$$

其中 $|C|$ 是同义词类 C 中项的个数，$w_{i,C}$是预先算好的项-同义词类对 $[k_i, C]$ 的权重。这样可以用这个平均项权重来计算同义词类的权重 w_C：

199

$$w_C = \frac{wt_C}{|C|} \times 0.5$$

上述的权重公式已经通过实验验证了，并产生了好的结果。

四个测试文档集（ADI、Medlars、CACM 和 ISI，这些文档集的详情参见 4.4 节）上的实验结果表明，使用由完全链接算法构建的全局分析可以在检索质量上产生一致的提高。

这个方法的主要问题是参数 TC、NDC 和 MIDF 的初始化。阈值 TC 依赖于文档集，且难于正确设置。设置 TC 时总是需要对聚类层次进行观察。因为高 TC 值可能会产生由少数几个项组成的同义词类，而低 TC 值可能会生成很少几个同义词类，所以要格外小心。当 TC 已经确定时，参数 NDC 可以更容易地确定。然而，参数 MIDF 的确定可能是困难的，并且也需要仔细地考虑。

5.7 趋势和研究问题

在相关反馈中的一个主要问题是如何大规模地整合反馈信息，也就是那些可用于提高大量不同的查询检索结果的信息。这在 Web 上尤为重要，因为查询覆盖了大量不同的主题。为了让相关反馈有效，就要在不干扰用户的情况下搜集反馈信息，这意味着 Web 上自然的用户反馈信息源是用户在搜索引擎结果上的点击。因此，一个主要的研究趋势是基于用户点击修改查询的研究。

局部分析技术是有趣的，因为它们利用了由查询提供的局部上下文。就这一点而言，它们看上去比全局分析技术更合适。而且，在文献中已经报告了许多积极的结果。然而，把局部分析技术应用到 Web 中需要进一步探索。由于需要在提交查询时处理文档，所以主要的困难是在搜索引擎上的计算代价。因此，一个和相关性有关的问题是搜索引擎要开发加快查询处理的技术。实际上，即使只考察正常的查询处理，这个问题也是值得关注的，因为搜索引擎为了能在经济上得到保障，就需要处理尽可能多的查询。

将局部分析、全局分析和用户点击相结合是当前重要的研究问题。而且，让用户可视化地探索文档空间，提供他一些线索，从而帮助表达查询似乎是一个值得期待的研究方向。即使研究人员和实践人员在这个领域没有获得一致的进展，考虑到用户界面的设计，积极的结果可能变成一个转折点，可能会吸引广泛的关注。

5.8 文献讨论

查询扩展方法的研究历史很长。尽管近年来扩展方法是否成功一直受到争议（该方法提高了召回率，但没有提高精度），但是目前的统计结果表明查询扩展是一项有用的技术。实 200
际上，现代搜索引擎经常把查询扩展技术应用到查询流中，从而推荐查询重构。

把和用户查询紧密联系的索引项用于查询扩展的早期工作是由 Maron 和 Kuhns[1093] 在 1960 年开展的。经典的技术是把向量模型中的查询扩展和索引项再赋权结合起来，这是由 Rocchio 在 1965 年提出的（使用了 Smart 系统 [1408] 作为测试平台），并之后发表在 [1375]。Ide 继续了 Rocchio 的研究，提出了索引项再赋权的变体公式 [806]。

概率模型是由 Robertson 和 Spark Jones[1365] 在 1976 年引入的。一份彻底的、令人欣赏的关于这个模型的讨论可以在 van Rijsbergen[1624] 的书中找到。Croft 和 Harper [452] 建议初始搜索应该使用不同的计算方法。1983 年，Croft[450] 提出扩展概率模型公式来包含文档内频率，并引入了参数 C 和 K。

由于概率模型没有提供扩展查询的方法，因此查询扩展是单独进行的。1978 年，Harper 和 van Rijsbergen[708] 使用了基于最大生成树的索引项聚类技术来为概率查询扩展挑选索引项。两年后，他们也引入了一个新的相关权重公式，称为 EMIM[707]，用于他们的查询扩展技术中。1981 年，Wu 和 Salton[1722] 使用相关反馈（采用概率公式）来对从相关文档中抽取的索引项再赋权，并用这些索引项来扩展查询。实验结果表明检索质量有所提高。

在5.3节中，我们对向量和概率模型中关于用户相关反馈的讨论是基于4个来源：Salton和Buckley[1410]的论文、van Rijsbergen[1624]的书、Salton和McGill[1414]的书以及Harman[700，701]中的两章。一项著名的关于相关反馈方法的比较研究是由Buckley和Salton在[1411]中提供的。其他关于相关反馈方法的文献可以在[1，29，289，290，1758]中找到。

通过点击进行显式反馈是近来令人激动的研究领域，这是由Joachims和他的助手[245，841，842，844，845，846，1320，1321，1322，1323]持续深入的工作推动的。他们的工作利用了之前对象排序学习的工作[407，591]，且激励了近来在使用点击提高Web排序的工作，例如Agichtein等人[17，18]的工作以及其他[580，1462，1687]。

对于自动的查询扩展，Lesk[1004]在Smart系统中尝试了不同的索引项聚类技术，却没有得到积极的结果。接着，Spark Jones和Barber[1509]，以及Minker、Wilson和Zimmerman[1137]也没有在索引项全局聚类上观察到提高。这些早期的研究结果给人留下这样的印象：基于全局分析的查询扩展不是一个有效的技术。然而，更多最近的研究结果表明情况不是这样。实际上，由Voorhees[1645]、Crouch和Yang[457]，以及Qiu和Frei[1309]获得的结果表明基于全局分析的查询扩展可以一致地产生更好的检索质量。

我们对于局部聚类的查询扩展的讨论是基于Attar和Fraenkel[78]在1977年的早期工作。局部上下文分析的思想是最近由Xu和Croft[1732]在1996年引入。采用全局相似度
201 同义词典在查询扩展上的讨论以Qiu和Frei的工作[1309]为基础。最后，关于使用全局统计同义词典的查询扩展的讨论是基于Crouch和Yang的著作[457]，后者受到了早期在1975年Salton、Yang和Yu[1417]的索引项区分度理论的影响。有关基于概念的Web查询扩展的讨论可以参见[572]。

由于查询扩展经常是基于某些形式的聚类，因此我们的讨论涵盖了某些聚类算法。更多关于文本聚类算法的讨论可以在第8章中找到。然而，我们的目的不是要提供一个用于信息检索的聚类算法的完整综述。这样的综述可以在Rasmussen的作品[1334]中找到。

相关反馈的思想在其他领域也获得了成功应用，例如基于内容的图像检索。其中，用户反馈信息是以图像正例和负例形式提供的。基于内容的图像检索中的相关反馈技术[435，
202 1058，1116，1393，1394，1395，1396，1784]用这些信息来估计初始查询。

第6章

文档：语言及属性

——与 Gonzalo Navarro 和 Nivio Ziviani 合著

6.1 介绍

文本是人类用来交流知识的主要记录方式，随着历史的进程，它们被写在石头、木头、动物的皮革、草纸或米纸的表面。如今在地球的每一个角落，文本以众多形式和语言写在纸上。文本被记录在包括打印格式或数字格式等多种媒介中，但是识别出它的单元是什么仍然不是一件容易的事。为了解决这个问题，我们一般将文本信息的基本单元定义为*文档*（document）。这个定义是非常宽泛的，比如一篇文档可能指的是一篇科研论文、一本书、一本手册、字典的一个条目、法官对某个案件的一个判决、汽车部件的说明，甚至是规模更大的文本的一部分，比如一个或多个段落的序列（在信息检索中这被称为*文本段落*（text passage））。就物理表示而言，一篇文档也可以是任何打印的或者数字表示的物理单元，如一个文件、一封电子邮件或者一个网页。

文档有特定的语法和结构，这往往是由文档的用途或者创建者来决定的。文档也包含了特定的语义，它是由文档的作者决定的，而文档的作者也不一定就是文档的创建者。另外，文档也可能包括和它关联的展示样式，用来指定文档应该怎样被显示或者打印。展示样式是与文档的语法和结构相关联的，并且根据特定的应用（如 Web 浏览器）定制。图 6-1 描述了所有这些关联。

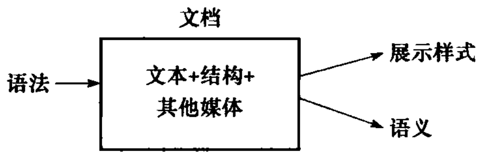


图 6-1　文档的特征

文档的*语法*（syntax）用来表示结构、显示样式、语义，甚至外部行为。在很多情况下，这些元素中的一个或者多个隐含在文本中或者被一同提供。比如，章节等结构元素就有固定的格式 203
样式。而且，文档的语义也是和它的用法相关联的。例如，Postscript 指令就是为绘画而设计的。

文档的语法可能会隐含地出现在它的上下文中，也可能会直接显式地体现在简单的说明性语言甚至是编程语言中。例如，很多编辑格式就是说明式的，而 TeX 文档就使用了一种强大的排版语言。尽管强大的语言往往比直接分析数据本身更加容易，但是将使用那种语言的文档转换为另外一种格式会变得非常困难。

很多语法的语言是特定的、专用的，但是开放的、通用的语言会更好，因为这样的文档可以在不同应用之间传递并且更加的灵活。文本也可用自然语言书写，尽管自然语言的语义会更难利用计算机自动解释。现在的趋势是利用能够表达出文档的结构、格式和语义信息，同时无论对于人类还是计算机都是可读的语言。标准通用标记语言（Standard Generalized Markup Language，SGML）试图平衡在上面提到的问题，我们将在随后的章节中涉及这部分内容。

大多数的文档都有特殊的版式样式。然而，新的应用程序都在力图使用外部格式，从而使得表示的信息和样式独立开来，反之亦然。展示样式可以嵌入在文档中，比如 TeX 或者富文本格式（Rich Text Format，RTF），并且可以利用宏（macro）来辅助（例如，LaTeX

就是 TeX 的一个宏）。在大多数情况下，样式是由文档的作者来定义的。然而，读者可能会决定某些样式特征，比如通过 Web 浏览器的设置选项。文档的样式定义了这个文档如何在计算机窗口中或者在纸上展示，而且也会包括对其他媒体，如音频和视频的处理。

搜索引擎查询，即使它们和文档并不完全相同，但也可以看成是由词语序列或者句子组成的短文本片段。然而，查询的特征不同于普通的文本，因此如何解释它们是非常重要的。此外，由于词语的多义性，查询的语义通常是模糊的，因此从查询中推测出用户的意图也是不容易的。

在本章中，我们将讨论元数据、文本特征（如格式和自然语言的统计特性）和用来描述文本结构、显示样式和语义的语言。之后，我们将讨论文本属性，怎样预处理和组织文档，
204 以及文本压缩。

6.2 元数据

大多数文档和文本集都伴随元数据。*元数据*（metadata）是关于数据组织、各种数据域以及它们之间关系的信息。简而言之，元数据是“数据的数据”。例如，在数据库管理系统中，模式（schema）指定了一些元数据，例如关系的名称、每个关系的字段（field）或属性以及每个属性的定义域（domain）等。

伴随文本的元数据的通常形式，包括作者、出版日期、出版来源、文档长度（以页数、词数或字节为单位）以及文档类型（如书、文章或者备忘）。例如，都柏林核心元素集（the Dublin core metadata element set）[1675] 提出了 15 个字段来描述文档。根据 Archionini [1082] 的定义，我们把这种信息称为*描述元数据*（descriptive metadata），它们和文档的意义无关，和文档如何被创建相关。另一种元数据描述了文档内容的主题，它们称为*语义元数据*（semantic metadata）。大量的文档都使用这些元数据，并且它们的可用性也在不断增加。比如，所有在美国出版的书籍都分配有美国国会图书馆（Library of Congress）主题代码，很多期刊都要求作者在一个包含术语的封闭字典里选择几个关键词。比如，出现在 MEDLINE(参见第 4 章）系统中的生物医学论文都需要分配一个与病理学、解剖学、制药学等相关的主题元数据。为了标准化语义术语，很多领域都使用了特定的*分类体系*（taxonomy），分类体系一般都是一个由描述某些知识主题的术语组成的层次结构，将在第 8 章中进行介绍。

一个重要的元数据格式是在图书馆记录中使用最多的机器可读目录（Machine Readable Cataloging Record，MARC）。MARC 是由多个字段组成的，这些字段对应不同的书目摘要条目，例如标题和作者。MARC 有着非常特定的用途，我们将在第 16 章中讨论。在美国，一个特殊版本的 MARC 也在使用，称为 USMARC，它符合 ANSI/NISO Z39.2 标准，ANSI/NISO Z39.2 是书目信息交换（Bibliographic Information Interchange）的美国国家标准。根据 Z39.2 标准，USMARC 格式包括了必须应用在书目结构中的字段的定义和内容。Z39.2 标准是由美国国会图书馆维护的。现在也有了 XML 版本的 MARC(参见 6.4.3 节）。

随着 Web 上数据量的激增，人们开始尝试给 Web 文档添加元数据信息。在 Web 上使用元数据有很多的原因，比如编目需求（cataloging）（BibTeX 就是这种情况下一个非常流行的格式）、内容评级（content rating）（比如，为了防止儿童阅读到不良文档）、知识产权保障、数字签名鉴定、隐私级别鉴别（用于文档的准入控制），以及实现用于电子商务的应用。就这一点而言，互联网内容选择平台（Platform for Internet Content Selection，PICS）使元数据与互联网内容相关联，并应用于内容评级、代码签名（code signing）和隐私控制。

事实上，PICS平台已经应用于多个内容审查服务和过滤软件。

Web元数据最重要的标准是资源描述框架（Resource Description Framework，RDF）[1660，983]，它使不同的应用程序共同工作成为可能。人们可以利用这个框架描述Web资源，以便更加容易地自动处理信息，在6.4.4节中，我们会更详细地讨论它。RDF并没有 205
设定任何特殊的应用或语义领域。它由结点和附在结点上的属性-值对的描述所构成。结点可以是任意的Web资源，也就是，统一资源标识符（Uniform Resource Identifier，URI），URI包括用于指定网页的统一资源定位符（Uniform Resource Locator，URL）。属性是结点的性质，值是文本串或者其他结点（Web资源或者元数据示例）。为了描述属性的语义和领域，可以使用都柏林核心元数据URL(Dublin Core library metadata URL)。对于内容评级和数字签名的应用，需要给定其他的预定义的元数据词汇表。

6.3 文档格式

6.3.1 文本

随着计算机的出现，文本需要用二进制数字进行编码。最初的编码方案是EBCDIC和ASCII，它们都是利用7位对字母表中的每个字母进行编码。之后，ASCII被标准化为8位(ISO-Latin)，使之可以适用于多种语言，包括重读符号以及变音符（diacritical）等符号。然而，ASCII码并不适合中文或日文汉字等东方语言，它们中的每一个符号可能代表一个概念，因此有成千上万的符号。针对这种情况，出现了16位的编码Unicode(ISO 10616)[1617]。除了字符外，对于文本并没有一种单一的标准格式。

过去，信息检索系统需要将文档转换为内部格式。然而，这样处理会有很多缺点，如不能对文档文本直接访问和直接修改。为了能够处理不同内部格式的大量文本，检索系统用过滤器来过滤大多数流行的文档格式，特别是那些文字处理软件，如Word和FrameMaker。即便这样，如果格式是专用的并且它的细节没有公布，那么过滤器就可能不能处理。但是就像TeX文档一样，如果语法是完全基于ASCII码的，那么就不存在问题。也就是说，用人们可读的ASCII码编码的文档有更强的可移植性且更容易修改，因为它们可以用各种不同的应用程序进行编辑。

人们也为文档交换设计了其他的文本格式。其中一个应当提及的是富文本格式（Rich Text Format，RTF），它是基于ASCII码的语法并且被Word系列的文本处理器使用。其他一些重要的格式为显示和打印文档而开发。其中最流行的就是便携文档格式（Portable Document Format，PDF）和Postscript，Postscript，是一种强大的编程式绘画语言。还有一些交换格式用来编码电子邮件，多用途互联网邮件扩展（Multipurpose Internet Mail Exchange，MIME）是一个很好的例子。MIME支持多种字符集、多种语言以及多种媒体。

为了减少存储空间，很多文件也需要压缩。文本压缩将在6.8节中详细地讨论，但是我们这里提及几个最流行的压缩软件以及它们对应的文件格式。它们包括Compress(UNIX)、ARJ(PC)和ZIP(如UNIX中的**gzip**以及Windows中的**Winzip**)。另外一些工具允许将二进制文件（特别是压缩过的文本）转换为ASCII码文本，从而使得文件可以通过一个只有7位的通信线路传输。比如**uuencode/uudecode**和**binhex**就是这类工具。 206

6.3.2 多媒体

多媒体通常用来处理源自不同类型媒介的不同类型的数字化数据。多媒体中最常见的媒

介是文本、声音、图像和视频（视频就是一个动态的图像序列）。由于上面四种媒介的数字化数据在容量、格式和处理要求（比如处理视频和音频数据有一定的实时性要求）上都有很大的不同，因此存储每种媒介都需要不同的格式。

相对文本格式而言，大多数的多媒体格式是部分二进制的，因此只能由计算机来处理。同时，显示样式也几乎完整定义，但在一些空间或时间属性上或许会有些例外。这里我们简要地介绍多媒体的主要格式和数据类型，关于如何在 Web 上使用它们将在第 11 章和第 17 章中讨论。

图像格式：GIF、JPEG、TIFF、PNG

图像格式有很多种。最简单的格式就是用位图（或者基于像素的表示）直接表示，比如 XBM、BMP 或者 PCX[⊖]。然而，这些格式占用太多空间。例如，典型的计算机屏幕中每个像素有 256 种颜色，描述一屏内容就需要 1MB 空间。实际上，图像存在很大的冗余并且可以被高效地压缩。因此，最流行的图像格式都进行了压缩，比如 Compuserve 公司的图像交换格式（Graphic Interchange Format，GIF）。GIF 格式适用于黑白图像，以及只包含少量颜色或者灰度（256 阶灰度）的图像。为了取得对高分辨率图像的更高压缩比，人们设计出了有损压缩算法。也就是说，一个压缩的图像不能解压缩并恢复到原始的图像。联合图像专家组（Joint Photographic Experts Group，JPEG）提出一种格式，这个格式尝试去除图像中对人眼影响很小的部分。这个格式是参数可调的，在这个意义上来说，图像的损失可以调整。

另一种常见的图像格式是标签图像文件格式（Tagged Image File Format，TIFF）。这个格式用于在不同的应用程序和计算机平台之间交换文档。TIFF 有一个元数据字段并支持压缩和不同数量颜色的图像。另外一个格式是 Truevision Targa(TGA）图像文件，这个格式与视频游戏平台相关。还有更多的图像格式，很多是和特殊的应用相关，比如传真（二值图像格式 JBIG)、指纹采集（高精度的压缩格式，如 WSQ）和卫星图像（高分辨率和全色彩图像)。1996 年，出现了一种在互联网上使用的位图图像格式：便携式网络图像（Portable Network Graphics，PNG)。这个格式已经成为 Web 上的事实标准。

为了能够适当地存储，音频首先需要数字化。最常见的小段数字音频格式是 AU、MIDI 和 WAVE。MIDI 格式是在电子乐器和计算机之间交换音乐的标准格式。对于音频库，
207 也使用像 RealAudio 或者 CD 格式等一些其他格式。

对于动画或动态的图像（类似于视频或者电视）也有多种格式，但是这里我们只介绍最流行的几个。最主要的一个是动态图像专家组（Moving Pictures Expert Group，MPEG）格式，这个格式与 JPEG 相关。MPEG 的原理是在固定间隔中只对和给定的基图像之间的差异进行编码。用这种方式，MPEG 受益于任何视频都有的时序图像冗余。更高的质量可以用更多的帧数（frame）和更好的分辨率来获得。MPEG 指定了不同的压缩等级，但是通常不是所有的应用都支持所有这些等级，请参阅第 14 章中的详细介绍。这个格式同时包含了与视频相关的音频信号。其他的视频格式包括 AVI、FLI 和 QuickTime。AVI 支持压缩(CinePac)，这点和 Apple 公司开发的 QuickTime 是相同的。和 MPEG 相同，这些格式也都包含了音频信息。

6.3.3 图形和虚拟现实

对于三维图形也有很多种格式。尽管这个主题并不是和信息检索完全相关，但为了讨论

⊖ PCX 的压缩版本利用了游程编码（run-length encoding)。

的完整性，我们也包含了一些相关的内容。尽管有很多的提议，但是我们这里的侧重点主要放在 Web 上。

为了公开交换结构化图形对象以及相关属性，人们定义了计算机图形元文件（Computer Graphics Metafile，CGM）标准（ISO 8632）。CGM 定义了一个二维数据交换标准，这个标准使得图形数据以与设备无关的方式在不同的图形设备、应用和计算机系统中存储和交换。CGM 是一个结构化的格式，它可以表示矢量图形（vector graphics）（比如多线段或者椭圆）、光栅图形（raster graphics）以及文本。尽管最初 CGM 是一个矢量图形格式，但它已经扩展了光栅能力并提供了一个非常有用的格式可以综合光栅和矢量图形。CGM 元文件是一些元素的集合。这些元素可能是图形的几何部件，比如多线段（polyline）或者多边形；这些部件的外观；怎样解释特定的元文件或者特定的图片。CGM 标准指定了哪些元素可以出现在元文件的哪些位置。

虚拟现实建模语言（Virtual Reality Modeling Language，VRML）（ISO/IEC 14772-1）是描述交互式 3D 物体和世界的文件格式，也是 Silicon Graphics 公司 OpenInventor 文件格式的一个子集。VRML 还是 3D 图形和多媒体的通用交换格式。VRML 可以用在很多应用领域中，比如工程和科学可视化、多媒体展示、娱乐与教育、网页以及共享虚拟世界。VRML 已经成为 Web 事实上的标准建模语言。

在 6.4.3 和 6.4.5 节中，我们将涉及与图像和多媒体相关的标记语言（markup language）。

6.4 标记语言

标记被定义为额外的文本语法，用来描述格式行为、结构信息、文本语义和属性等。例如，TeX(一个流行的文本排版软件）的格式命令就可以看做是标记。然而，正式的标记语言是更加结构化的，这些标记（mark）被称为标签（tag），并且为了防止混淆会有一个开始和结束的标签包围着被标记的文本。标准的标记元语言是已经提过到的 SGML。SGML 的 208
一个重要的子集就是 Web 的元语言——可扩展标记语言（eXtensible Markup Language，XML）。Web 使用的标准标记语言是超文本标记语言（Hyper Text Markup Language，HTML），它也是 SGML 的一个特例，但是最新的版本也和 XML 兼容了。所有这些语言，包括它们的实例会在随后进行讨论。

6.4.1 SGML

标准通用标记语言（Standard Generalized Markup Language，SGML）（ISO 8879）是 Goldfarb[633] 领导的小组基于 IBM 的早期工作开发的一个标记文本的元语言。SGML 提供了一套基于标签来定义标记语言的规则。每个 SGML 实例都包括文档的结构描述，称为文档类型定义（document type definition）。因此，SGML 文档通过下面两个部分进行定义：1）文档结构的描述；2）用描述结构的标签所标记的文本。后面会阐述标签对应的语法。

文档类型定义用来描述和命名构成文档的那些片段并定义这些片段是如何彼此相关的。一部分定义可以用 SGML *文档类型声明*（document type declaration）或 DTD 指定。另一部分不能用 SGML 正式地表达，比如元素和属性的语义或者应用的协议，但可以利用注释非正式地表达。这意味着所有将 SGML 标签应用到文档的规则是定义的一部分，并且可以表示为 SGML 语法的那部分在 DTD 中表示出来。DTD 没有定义标签的语义（也就是，意义、

描述和行为）或预期用途。然而，有些语义信息可以内嵌在 DTD 的注释中，而更完整的信息通常可以在单独的文件中描述。这个单独的文件通常描述元素（或称数据的逻辑片段）以及这些数据片段的属性及其他信息。例如，两个标签可能有相同的命名但是在不同的应用中可能会有不同的语义。

标签用尖括号（**<tagname>**）来表示。标签用来指定文档片段的开始和结束，好比文学文本中的引号。属性在元素的开始指定，在结束标签名字前面加一个斜线（如**<\tagname>**）来指明标签的结束。例如，标签**<\author>**可以用来指明“作者的姓名”这个元素，这个元素以斜体形式出现并产生一个指向传记的链接。标签属性在尖括号里定义，属性跟在标签名后面以 **attname＝value** 的语法形式出现。

图 6-2 给出了一个简单的 DTD 定义以及一个使用它的文档。尽管我们这里不打算讨论 SGML 的语法，但是我们会提供一个对例子的概要描述，使读者可以理解其主要思想。每

```
<!--SGML DTD for electronic messages -->

<!ELEMENT e-mail             - - (prolog, contents) >
<!ELEMENT prolog             - - (sender, address+, subject?, Cc*) >
<!ELEMENT (sender | address |
          subject | Cc)      - O (#PCDATA) >
<!ELEMENT contents           - - (par | image | audio)+ >
<!ELEMENT par                - O (ref | #PCDATA)+ >
<!ELEMENT ref                - O EMPTY >
<!ELEMENT (image | audio)    - -  (#NDATA) >

<!ATTLIST e-mail
          id         ID                  #REQUIRED
          date_sent  DATE                #REQUIRED
          status     (secret | public )  public >
<!ATTLIST ref
          id         IDREF               #REQUIRED >
<!ATTLIST (image | audio )
          id         ID                  #REQUIRED >

<!--Example of use of previous DTD-->
<!DOCTYPE e-mail SYSTEM "e-mail.dtd">
<e-mail id=94108rby date_sent=02101998>
 <prolog>
 <sender> Pablo Neruda </sender>
 <address> Federico Garcia Lorca </address>
 <address> Gabriel Garcia Marquez </address>
 <subject> Pictures of my house in Isla Negra
 <Cc> Ernest Hemingway </Cc>
 </prolog>
 <contents>
   <par>
   As promised in my previous letter, I am sending two digital
   pictures to show you my home and the view full of light of
   the Pacific Ocean from my bedroom (photo <ref idref=F2>).
   </par>
   <image id=F1> "photo1.gif" </image>
   <image id=F2> "photo2.jpg" </image>
   <par>
   Regards from the South, Pablo.
 </contents>
</e-mail>
```

图 6-2 结构化电子邮件的 DTD 以及使用实例

个 ELEMENT 定义了一个标签和对应的名字。后面的两个字符表明开始和结束的标签是强制的（-）或者是可选的（o）。例如 **prolog** 的结束标签是必需的，而 **sender** 的结束标签并不是必需的。接着，用正则表达式风格的语法定义标签的内部内容，如“,”表示连结，“|”表示逻辑或，“?”表示前面的元素 0 次或者 1 次出现，“ * ”表示前面的元素 0 次或多次出现，“+”代表之前的元素 1 次或者多次出现。内容标签可以由其他标签内容、ASCII 字符（PCDATA）或者二进制数据（NDATA）组合而成；或者是空（EMPTY）。可能的标签属性由标签名字表示的属性列表（ATTLIST）中给定，后面是每个属性的名字、属性的类型以及它是否必须存在（否则，将会给定一个默认值）。当 SGML 文档实例使用 DTD 时，处理这个数据的各种工具知道哪些是正确的标签以及它们的组织方式。 209~210

这个文档描述一般并没有指定文档应该怎样打印在纸上或者显示在屏幕上。因为 SGML 将内容和格式分离开来，我们可以创建很好的数据模型，而不用考虑如何描述它的格式，也没有一个标准的格式来输出数据。因此，作为输出文档指南的输出规范（output specification）经常被加入到 SGML 文档中。为此，人们设计了输出规范标准，如文档样式语义和规范语言（Document Style Semantic Sepcification Language，DSSSL）以及格式输出规范实例（Formatted Output Sepcification Instance，FOSI）。两个标准都定义了格式信息和 SGML 文档实例的关联机制。它们是 SGML 文档系统的一部分，例如它们可以定义某个标签标注的数据应该以斜体输出。

一个重要的 SGML 应用是文本编码先导计划（Text Encoding Initiative，TEI）。TEI 是由美国多个人类学和语言学的相关协会在 1987 年开始的一个合作项目。这个项目的主要目的是为学术研究和工业应用中的电子文本的准备和交换提供指导。除了指导，TEI 还通过 SGML DTD 提供了多个文档格式。其中最常用的一个是 TEI Lite。TEI Lite DTD 可以单独使用或者与全部的 TEI DTD 文件一起使用。

6.4.2　HTML

超文本标记语言（HyperText Markup Language，HTML）也是 SGML 的一个实例。HTML 在 1992 年提出并在过去的几年里迅速发展，1999 年年底发布的 4.01 版是其最新的版本。现在 HTML 仍然不断改进以解决它的多种局限，例如不允许数学公式的定义。即使有这样的局限，但在 Web 上的大部分文档仍然是以 HTML 存储和传输的。虽然有一些历史原因，但我们也应该记住 HTML 是一种简单的语言，适合超文本、多媒体以及小而简单的文档显示。

HTML 基于 SGML，虽然 HTML 也有一个 HTML 文档类型定义（Document Type Definition，DTD），但是大多数的 HTML 实例都不会显式地引用 DTD。HTML 标签遵循所有的 SGML 约定，也包含了格式指南。

HTML 文档里可以嵌入其他媒体，比如不同格式的图像或者音频。HTML 也有一些元数据的字段，可以用于不同的应用和目的。如果在页面中加入程序（比如 JavaScript 程序），即通常所说的动态 HTML（或 DHTML）。注意不要把这个概念和 Microsoft 公司的一个协议（也称为动态 HTML）混淆，这个协议是一个用来读取和操作 HTML 文档的应用程序编程接口（Application Programming Interface，API）。图 6-3 给出了一个 HTML 文档的实例以及它在 Web 浏览器中的输出。

```
<html>
<head>
<title>HTML Example</title>
<meta name=rby content="Just an example">
</head>
<body>
<h1>HTML Example</h1>
<p>
<hr>
<p>
HTML has many <i>tags</i>, among them:
<ul>
<li>links to other <a href=http://www.w3c.org/>pages</a> (a=anchor),
<li>paragraphs (p), headings (h1, h2, etc), font types (b, i),
<li> horizontal rules (hr), indented lists and items (ul, li),
<li> images (img), tables, forms, etc.
</ul>
<p>
<hr>
<p>
<img align=left src="at_work.gif">
This page is <b>always</b> under construction.
</body>
</html>
```

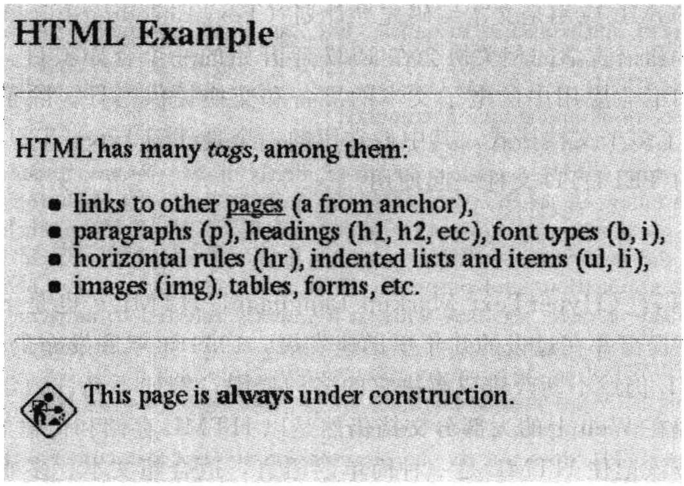


图 6-3 HTML 文档的例子以及在浏览器中的效果

因为 HTML 没有固定文档的展示样式，所以层叠样式表（Cascade Style Sheet，CSS）在 1997 年引入。CSS 给作者、艺术家以及印刷商提供了一个强大的、可操控的创建视觉效果的方式，以增强 Web 上的 HTML 网页的美感。样式表（style sheet）可以一个紧跟另一个（称为层叠式）地使用，以定义 HTML 页面的不同元素的显示样式。样式表将展示信息
211～212 与文档内容分离开来，简化了网站的维护，加强了网页的可访问性，并且加快了 Web 的速度。另一方面，在当前的浏览器中对 CSS 的支持还是有限的。另外一个缺点是两个样式表并不一定要求一致或完整，因此样式的结果可能并不好，特别是在颜色方面。CSS 用来在作者和读者对展示方面的不同预期中建立平衡。然而，它却不能确定在什么情况下由作者还是读者来定义展示样式。

HTML 的发展意味着它不但支持向后兼容而且支持向前兼容，因为人们可能也需要利

用旧的浏览器来看新的文档。HTML 4.0 被设计成三种：严格的（Strict）、过渡的（Transitional）和框架结构的（Frameset）。Strict HTML 只关心与展示无关的标签，而将所有的展示信息都留给 CSS 处理。Transitional HTML 利用所有的页面展示特征使得其可以被不支持 CSS 的旧浏览器读取。Frameset HTML 用于需要将浏览器窗口分割为两个或者更多个框架的时候。HTML 4.0 包含了对样式表、国际化、框架、更丰富的表格和表单，以及为残疾人士设计的辅助功能选项的支持。

典型的 HTML 应用采用了一个小的固定标签集合，这些标签符合 SGML 规范。小的固定标签集合使得用户可以不把语言规范放在文档中，也使得创建应用程序变得容易，但是这个优点的代价是 HTML 在某些重要方面的严格限制。特别是，HTML：

- 不允许用户为了参数化或语义地描述它们的数据而指定自己的标签和属性。
- 不支持用于表示数据库模式或面向对象层次结构的嵌套结构规范。
- 不支持允许应用程序检查数据结构合法性的语言规范。

对比 HTML，一般的 SGML 应用能支持任意复杂度的 SGML 语言规范，从而使得扩展、结构表示以及校验都成为可能，而这些都是 HTML 所缺少的。为了能够处理大量的复杂文档，从而管理大型信息资源库，SGML 允许用户为他们的文档定义自己的格式。然而，完整的 SGML 包含了很多对于 Web 应用没有必要的额外特性，并且证明对于当前的 Web 浏览器开发商其“成本/效益”比并不具有吸引力。所有这些因素导致了 XML 的发展，XML 是一种比较简单的元语言，将在下一节中讲述。

HTML 的发展有两条不同的路线。第一条是 XTHML 1，这是一个和 XML(见 6.4.3 节）兼容的 HTML 版本，其上一个推荐版本在 2002 年发布。之后，提出了 XHTML 2(上一个草案是从 2006 年开始的)。XHTML 2 创建了一个体系架构，并与 XML 相兼容，向成为 Web 技术的主要语言迈进了一大步。另一条路线是最近被提出的 X/HTML 5，它是 HTML 4 和 XHTML 1 的扩展。它是一个增量式的进步，目的是解决 HTML 4 和 XHTML 1 的很多弊端。特别是，X/HTML 5 可以用做 HTML 或者 XML，这取决于标记在当前如何被使用。 213

6.4.3 XML

可扩展标记语言（eXtensible Markup Language，XML）是 SGML 的一个精简集。也就是说，XML 和 SGML 都是一种元标记语言。XML 允许使用可读的语义标签，这些标签也是机器可读的。因此，XML 使得开发和部署新的特定标记变得容易，从而能够自动地编写、分析和处理网络数据。在某种意义上，XML 实现了很多原本用 Java 脚本或者其他程序接口实现的功能。

XML 不像 HTML 那样强加了很多限制。但是，XML 提出了一个更严格的标记语法，这个语法在处理阶段非常重要。在 XML 中，结束标签不能省略。对于没有任何内容的标签，比如 BR 和 IMG，也需要在尖括号结束前标上“/”以特殊标识。XML 也区分大小写，所以 **img** 和 **IMG** 是不同的两个标签（在 HTML 中不是这样）。另外，所有的属性值都必须在引号中。这也意味着我们在不知道标签信息的情况下可以比较容易地解析 XML。特别是，当 DTD 是可选的。如果没有 DTD，分析结束后才会获得标签。与 SGML 相比，XML 在语法上有些不同并有很多限制。列出所有这些不同超出了本书的范围，但是图 6-4 展示了一个没有 DTD 的 XML 的例子，这个 XML 文档基于之前的 SGML 电子邮件 DTD(参见图 6-2)。RMD 属性表示请求标记声明（Required Markup Declaration），指示是否必须使用 DTD(在

这个例子中没有使用 DTD）。其他可能的值是 **INTERNAL**，意味着 DTD 文件在文档内部；以及缺省值 **ALL**，它允许像 SGML 一样使用外部资源作为部分（或者全部）的 DTD。

```
<?XML VERSION="1.0" RMD="NONE" ?>
<e-mail id="94108rby" date_sent="02101998">
 <prolog>
 <sender> Pablo Neruda </sender>
 <address> Federico Garcia Lorca </address>
 <address> Gabriel Garcia Marquez </address>
 <subject> Pictures of my house in Isla Negra</subject>
 <Cc> Ernest Hemingway </Cc>
 </prolog>
 <contents>
   <par>
   As promised in my previous letter, I am sending two digital
   pictures to show you my home and the view full of light of
   the Pacific Ocean from my bedroom (photo <ref idref="F2"/>).
   </par>
   <image id="F1" ref="photo1.gif" />
   <image id="F2" ref="photo2.jpg" />
   <par>
   Regards from the South, Pablo.
   </par>
 </contents>
</e-mail>
```

图 6-4 与图 6-2 相似的不带 DTD 的 XML 文档

XML 允许用户定义新的标签，定义更加复杂的结构（如利用与 SGML 相同规则的无限嵌套）并且加入了数据校验能力。尽管 XML 比较新，但仍然有一些关于 XML 如何改变或影响互联网应用的讨论。XML 是 SGML 的一个简化版，它除去了很多难以实现的东西，因此对于大部分情况它和 SGML 是一样的。像之前提到的，XML 除去了 DTD 必须存在的限制，DTD 可以直接从数据中分析得到。删除 DTD 提高了应用程序文档的重要性。这对软件提供的功能有很大的影响。例如，如果一个 XML 编辑不使用 DTD，怎样帮助用户一致地标记文档呢？这些问题都应当在不久的将来得到解决。在标签名字之间有语义歧义的情况下，可以使用命名空间（namespace）来保证它的使用约定。

可扩展样式表语言（Extensible Style sheet Language，XSL）是 XML 对应的层叠样式表（Cascading Style Sheets，CSS)。XSL 用来对高度结构化和丰富数据的 XML 文档进行转换和样式设计。例如，利用 XSL 可以从文档中自动抽取目录。XSL 的语法利用 XML 定义。除了利用向文档中添加格式的功能外，XSL 还能够用来将 XML 转化为 HTML 和 CSS 文档。这类似于文字处理工具中的宏。

可扩展链接语言（Extensible Linking Language，XLink）是 XML 的另外一个扩展，也是用 XML 定义的。XLink 定义了不同类型的链接，包括外部链接和内部链接。特别是，任何元素类型都可以作为链接源，外部链接可以定义在不能修改的文档上。链接的行为也是一
214 般化的。链接对象可以嵌入或者代替文档。也有可能在不改变当前应用的情况下产生一个新的上下文（例如，在一个新的窗口中显示对象)。

另外一个扩展是 XML 指针语言（XML Pointer Language，XPointer)，XPointer 是对 URI 引用的片段标识符，这个 URI 引用可以定位互联网上 XML 类型的媒体资源。XPointer 基于 XML 路径语言（XML Path Language，XPath）（见 13.6.3 节)，用来标识 XML 文档的内部结构。它允许通过多个属性对层次结构的文档进行检查并选择内部片段，这些属性包

括元素类型、属性值、字符内容以及相对位置。

另外一个与 XML 相关的组件是文档对象模型（Document Object Model，DOM）。DOM 是一个与平台及语言无关的接口，它允许软件动态地读取并更新文档的内容、结构和格式。文档可以被进一步处理并且可以把处理结果合并到当前的页面。也就是说，DOM 提供了一个可互操作的类和方法的集合，例如 Java 等编程语言中的 HTML 和 XML 的对象。

XML 的重要应用包括：

- 数学标记语言（Mathematical Markup Language，MathML）：两个标签的集合，一个用于数学公式的显示，另外一个是数学表达式的含义。
- MARCXML：一个基于 MARC21 风格的书目 MARC 标准的 XML 模式。它是美国国会图书馆（US Library of Congress）开发的，并且被美国国会图书馆和其他组织 215 采用作为分享和网络获取书目信息的手段。由于它很容易被各种系统分析，因此它可以用做聚合格式。这个模式旨在以灵活和可扩展的方式使用户根据需求独特地使用 MARC 数据，它包括很多的组件，比如模式、样式表和软件工具。
- Web 服务描述语言（Web Services Description Language，WSDL）：一个基于 XML 的语言，用来描述网络服务。网络服务是指针对面向文档或者面向过程的信息的一系列操作代理。这些操作和信息被抽象地描述，并被具体的网络协议和信息格式约束，从而定义代理。扩展 WSDL 是为了使得代理及其信息描述与具体使用什么样的网络协议无关，如 SOAP、HTTP 或 MIME。
- 可缩放矢量图形（Scalable Vector Graphics，SVG）：使用 XML 描述二维图形和图形应用。SVG 可以作为网页浏览的交换格式，也可以作为交互式的多媒体平台，并且也可以是可持续 Web 的一个主要部分。一个有趣的特性是，在 SVG 中的文本并不只是图形，而是可以被搜索和选择的。最近，W3C 发布了 SVG 的一个子集 SVG Tiny，用于移动应用和其他的嵌入式多媒体系统。这个版本已经被广泛地应用到移动电话中。
- 墨水标记语言（Ink Markup Language，InkML）：一个用来表示电子墨水数据的 XML 数据格式，电子墨水数据是多模态系统中电子笔的输入。这个格式用来在设备和软件之间传递数字墨水数据，为手写体识别、签名认证和姿势识别存储手写痕迹。

XML 的发展预示着可分析的层次化对象模型将会在 HTML 的发展中扮演越来越重要的角色。下一代 HTML 可能会基于 XML 的标签集合，用来标识数学、同步多媒体和矢量图像（可能使用已经提到的基于 XML 的语言）。这也意味着对数据结构化和建模成为重点，而不是数据显示和布局。在第 13 章中，我们将详细地讨论 XML 检索。

6.4.4 RDF

资源描述框架（Resource Description Framework，RDF）最初被设计为元数据数据模型（在 6.2 节中提到）的一系列规范。然而，现在它已经成为采用一系列语法格式对 Web 资源的概念性描述（conceptual description）和建模的普遍方法。实际上，它已经是语义网（Semantic Web）事实上的标准语言。

RDF 数据模型与实体关系（Entity-Relationship）和类（Class）图等经典的概念建模方法类似。这个模型基于这样的想法，利用主语-谓词-宾语的三元组对资源进行陈述，特别是 Web 资源。主语表示资源，谓词是资源的属性并表示主语和宾语的关系。例如，在 RDF 中 216 表示“The Chilean flag has the red color”的概念利用了如下的三元组：主语是“the Chil-

ean flag"，谓词是"has the color"，宾语是"red"。

RDF语句集是一个带标签的、有向多边图。正因为如此，RDF比某些传统模型（如关系模型和本体模型）更适合表达特定类型的知识。RDF数据经常存储在关系数据库中，或者存储在称为三元组存储（Triplestores）、四元组存储（Quad-stores）的原生表示中。在四元组存储中，上下文信息（命名图）也被存储。

用来描述资源的方法是语义网（Semantic Web）活动的一个主要部分：这是Web的一个进化阶段，在这个阶段中，软件可以存储、交换和使用Web上发布的机读信息，这使得用户能够高效率和可靠地处理信息。RDF的简单数据模型和对抽象概念建模的能力也使得它可以应用到其他的知识管理应用中。

资源描述框架模式（The RDF Schema，RDFS）是一个扩展知识表示语言，它给出了用来描述本体的基本元素，也称为RDF词汇，用来构建RDF资源。很多RDFS组件包括在更具表达力的语言——互联网本体语言（Web Ontology Language，OWL）中。OWL是一系列用来创建本体的知识表示语言，也被W3C所认证。OWL建立在两个几乎完全兼容的语义上：1）OWL DL和OWL Lite；2）OWL Full。前一组基于描述逻辑（Description Logic），描述逻辑具有吸引人的并且容易理解的计算属性。OWL Full使用一个试图和RDFS保持兼容的语义模型。OWL定义的本体通常利用RDF/XML序列化。正如RDFS和OWL所证明的，附加的本体语言可以是基于RDF构建的（见第17章），正因为这个原因，OWL看做是语义网背后的一项基础技术。

6.4.5 HyTime

超媒体/基于时间的结构语言（Hypermedia/Time-based Structuring Language，HyTime）是定义多媒体文档标记的标准（ISO/IEC 10744）。HyTime是规定文档通用超媒体结构的SGML架构。根据SGML的指导原则，HyTime定义的结构与要表示的编码文档相互独立。作为一个架构，HyTime允许为每个使用HyTime结构的文档模型编写DTD，DTD指明了这些文档集如何修改这些结构用于它们特殊的表示需求。这个标准也提供了多种元DTD用来方便新的多媒体标记语言的设计。

直接使用HyTime表示的超媒体内容包括：

- 文档对象的复杂定位。
- 文档对象之间的关系（超链接）。
- 文档对象之间数值的、可度量的关联。

217 HyTime架构包括三个部分：基本的链接和定位架构、调度架构（从基本架构演变的）和生成架构（这是调度架构的一个应用）。基本架构处理超链接的语法和语义。对于最简单的超媒体表示，这些应该足够了。HyTime的调度模块定义了包括音乐和交互表示在内的复杂超媒体结构的抽象表示。它的基础技术是一个比较简单的技术：时间或空间轴上的对象容器序列。生成模块是一个调度架构的应用，它通过不同类型的"生成规则"，定义如何从已有调度中创建出新的调度。

HyTime并没有直接指定图形接口、用户导航、用户交互或者在时间轴或屏幕上的媒体的布局。文档处理是从HyTime结构中以样式表的方式来生成的，就像SGML文档那样。

HyTime的一个应用是标准音乐描述语言（Standard Music Description Language，SMDL）。SDML是一个表达音乐信息的架构，SDML用来单独或和其他媒体一起表示音乐信息，也支持多媒体时序信息。另外一个应用是交互文档元文件（Metafile for Interactive

Documents，MID)。MID 是一个基于 SGML 和 HyTime 的通用交换结构，它以最少的人工干预从不同的创作系统和结构中获得数据并在不同的表示系统中显示。

HyTime 的 XML 实现是同步多媒体集成语言（Synchronized Multimedia Integration Language，SMIL)。SMIL 是一个用于规划 Web 上多媒体展示的声明语言，可以控制不同对象出现的位置和时间。

6.5 文本属性

在本节中，我们将涉及不同的文本特征。首先，我们讨论如何度量文本的信息内容，并介绍几种不同的模型。我们同时也简要地讨论了怎样度量文本字符串或片段之间的相似度。

6.5.1 信息论

书面文本是用来交流信息的一个途径，因此总是包含一定量的语义内容。即使很难正式地得到给定文本中的信息量，但信息量和文本中符号的分布情况是相关的。例如，几乎只出现同一个符号的文本不会表达出太多的信息。我们可以利用这个思路给不同的符号赋予不同的位序列或编码。

这个理论是克劳德·香农（Claude Shannon）在著名的信源编码理论［1452］中建立的。他指出，在最优的编码方案中，一个以期望概率 p 出现的符号应该被赋予长度为 $\log_2 \frac{1}{p}$ 位的码字。最优编码的位数代表一个符号的信息量。因此，我们可以通过每个符号出现的概率和赋予每个符号的码字定义一个统计文本模型。 218

在这个文本模型下，文本 $T=t_1 t_2 \cdots t_n$ 中每个符号的平均信息量称为这个文本的熵，其定义如下：

$$E = \frac{1}{n}\sum_{i=1}^{n}\log_2\frac{1}{p_i} \tag{6-1}$$

其中 p_i 是这个模型赋给符号 t_i 的概率。值得注意的是 E 是通过概率计算的，因此它不仅是文本的属性，还是模型的属性。

在比较简单的情况下，这个模型给字母表符号 s_i 赋予一个概率 p_i，于是熵可以重写为

$$E = \sum_{s_i\in\Sigma}p_i\log_2\frac{1}{p_i} = -\sum_{i=1}^{\sigma}p_i\log_2 p_i \tag{6-2}$$

其中Σ是文本的字母表，并且 $\sigma=|\Sigma|$。在这个公式中，字母表中的 σ 个符号用二进制编码，因此熵的单位是位。例如，在 $\sigma=2$ 时，如果两个符号出现同样的次数，则熵是 1；而如果只有一个符号出现，则熵是 0。熵可以用来度量信息内容（或者信息的不确定性)，因此我们说在文本中的信息量可以用来它的熵来衡量。这个概念在文本压缩中也很重要。根据这个文本模型，熵就是文本压缩程度的极限。6.8 节将详细地讨论文本压缩。现在讨论我们所关心的是自然语言的文本模型。

6.5.2 自然语言建模

文本是由有限字母表中的符号组成，字母表中的符号可以分为两个不相交的子集：分隔单词的符号，通常称为分隔符，以及属于单词的符号。一个重要的观察是这些符号在文本中并不是均匀分布的。为了说明这一点，如果我们只考虑字母（a～z)，那么我们可以看到元音往往比大多数辅音出现的频率更高。例如，在英语中，字母“e”出现的频率最高。

一个比较简单的生成文本模型是二项式模型。在这个模型中，每个符号都以某个概率独立生成。这个模型当然是简化的，因为在自然语言中符号之间是有依赖关系的。例如，在英语中，字母“f”不能出现在字母“c”的后面，而且元音和某些辅音出现的频率更高。因此，一个符号的概率依赖于之前的一个符号。为了获得这些依赖关系，我们可以使用有限上下文或马尔科夫模型。该模型利用 1、2，或者更多个字母产生下一个符号。如果使用 k 个字母，我们说这是一个 k 阶模型（因此二项式模型可以看做是一个 0 阶模型）。我们可以把单词看做符号并应用这些模型，正如在介绍语言模型的第 3 章中所讨论的。例如，用圣经中词的分布统计出来的 5 阶模型生成的文本可能是有意义的（就是说，它可能在语法上是正确的），但是一定和原来的圣经是不同的。更复杂的模型包括，定义了正则语言的有限状态机
219 模型，以及定义了上下文无关及其他类型语言的语法模型。然而，找到自然语言的合适语法仍然是一个开放的、困难的问题。

另外一个重要的问题就是不同的单词是怎样在每个文档中分布的。一个近似的模型是齐夫法则（Zipf’s Law）[1794，649]，它统计文本中词汇出现频率（即出现次数）的分布。这个规则指出，出现次数最多的第 i 个词的频率 f_i 是第一个词的频率 f_1 的 $1/i^\alpha$ 倍，其中 α 是一个与文本相关的参数。也就是

$$f_i = \frac{f_1}{i^\alpha} \tag{6-3}$$

原始的齐夫法则使用 $\alpha=1$，并且当 $\alpha>1$ 的时候我们称之为广义齐夫法则。对于一个建立在 V 个单词的词汇表上的 n 个单词的文档，我们可以得到：

$$n = \sum_{i=1}^{V} \frac{1}{i^\alpha} f_1 = f_1 \times \sum_{i=1}^{V} \frac{1}{i^\alpha}$$

括号中的因子仅仅依赖文本参数 α 和 V，它称为 V 的 α 阶调和数 $H_V(\alpha)$，即

$$H_V(\alpha) = \sum_{i=1}^{V} \frac{1}{i^\alpha} \quad \text{那么} \quad f_1 = \frac{n}{H_V(\alpha)} \tag{6-4}$$

齐夫法则的一个直接推论是，出现最多的第 i 个词出现了 $n/(i^\alpha H_V(\alpha))$ 次。

图 6-5 的左侧展示了典型的频率分布，其中词是按照频率的降序排列的。α 的值依赖于文本。在最简单的情况下以及最原始的公式中，$\alpha=1$，因此 $H_V(\alpha)=O(\log V)$。然而，这个最简单的版本是非常不准确的，而 $\alpha>1$（更准确地说是在 1.5～2.0 之间）的情况更适合真实的数据 [65]。这种情况是幂律的一个例子（参见 11.3.3 节），但却非常不同，因为这里的分布更加不均衡，并且 $H_V(\alpha)=O(1)$。实验数据证明更好的模型是 $k/(c+i)^\alpha$，这里 c 是一个额外的参数，k 是为了使得所有的频率加起来等于 n，称为 Mandelbrot 分布 [1133]。

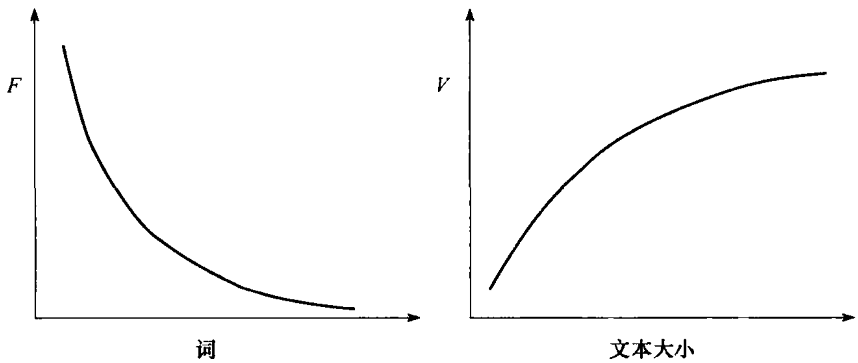


图 6-5 词频排序后的分布（左）和字典大小的分布（右）

既然词汇的分布是非常偏斜的（这是说几百个单词却占据了文本中出现的词的 50%），以非常高的频率出现的词，例如禁用词（stopword），应该被忽略掉。禁用词是那些在自然语言中并没有多少意义的词，因此应该被忽视掉（这也使得这些词不能被搜索到），例如 **"a"**、**"the"**、**"by"** 等。幸运的是，出现最多的词都是禁用词，因此文本中一半的词不需要考虑。这样，为自然语言文本建索引时空间开销会明显减少。例如，在 TREC-2 文档集（更详细地了解此文档集或其他文档集参见 4.4.1 节）中最多出现的词是 **"the"**、**"of"**、**"and"**、**"a"**、**"to"** 和 **"in"**。需要注意的是，从索引中去除禁用词可以使得索引简洁，但是这是有代价的，就是这个词不能在索引中被搜索到了。例如，如果禁用词都被从索引中删除，搜索 **"The Who"** 乐队时可能就会失败。然而，这是在 Web 上非常重要的查询类型，搜索引擎采 220
用了全文索引模式来避免这个问题，即对所有的词建立索引。

第三个问题是词在文档集内的分布。简单的模型假设在每个文档中每个词都出现了同样的次数。然而在实际中，这个假设并不成立。比较好的模型会采用一个负二项分布，即文档包含一个出现 k 次的词的概率 $F(k)$ 是，

$$F(k) = \binom{\alpha + k - 1}{k} p^k (1-p)^{-\alpha-k} \tag{6-5}$$

其中 p 和 α 都是依赖于这个词和文档集的参数。例如，对于布朗语料库（Brown Corpus）[583] 和词 **"said"**，可以得到 $p=9.24$ 和 $\alpha=0.42$[384]。之后的参考文献中还给出了从泊松分布衍生的其他模型。

第四个问题是文档中不同单词的数量。单词的集合称为文档的*词汇表*（vocabulary），记为 V。我们使用 Heaps 法则 [730] 来预测自然语言文本中词汇表大小的增长。这是一个非常精确的规则，它指出包含 n 个词的文本的词汇表的大小是 $V=Kn^\beta=O(n^\beta)$，这里的 K 和 β 都依赖于特定的文本。图 6-5 的右侧给出了词汇表大小随文档大小变化的情况。K 一般都在 10～100 之间，β 是一个小于 1 的正数。有些在 TREC-2 文档集上的实验 [65，116] 指出，β 最常用的值是在 0.4～0.6 之间的。因此，文档的词汇表随着该文本大小按近似其平方根的比例次线性增长。

值得注意的是一种语言中的不同词的集合是由一个常数限定的（例如不同的英语单词的数量是有限的）。然而这样的限制实在太宽泛了，以至于假设词汇表的大小是 $O(n^\beta)$ 比 $O(1)$ 更准确，虽然对于足够大的文本，这个数目应该是稳定的。另一方面，因为输入和拼写错误的原因，很多人坚持词汇表的大小会持续增长。 221

Heaps 法则也应用到文档集中，因为随着文本总量的增长，这个模型的预测也会变得更准确。而且，这个模型也可以应用到 Web 中（参见第 11 章）。

最后一个问题是词的平均长度。平均长度把文本的单词数量和字节数关联起来（不考虑标点和其他符号）。例如，在 TREC-2 文档集的不同子集中，词长的平均值非常接近 5 个字母，并且在每个子集中平均值的变化范围也很小（4.8～5.3 个字母）。如果我们除去禁用词后，词长的平均值会增长到 6～7 个字母。如果我们仅仅考虑在词汇表中的单词，那么平均长度会更大些，大概 8～9 个字母。这个值定义了词汇表的空间需求。

不变的平均词长是与 Heaps 法则不相符的，原因如下面所述。随着文本大小 n 的增长，Heaps 法则预测词汇表也会增长，这意味着用来表示所有不同的词的字母数将会增加。因此，越来越长的单词会随着文本的增长而出现。只有当相对较短的单词的出现足够普遍时（这是经常的事），平均长度才会保持不变。实际上，这个效果并不明显，因此可以假设平均词长是不变的，是与文本的大小相独立的。像在不同上下文中都被多次注意到的那样，平均

词长保持不变，这种在短单词和长单词之间的平衡可以通过有限状态机模型解释：1）空格的概率接近 0.2；2）空格不能连续地出现两次；3）26 个字母符合一个固定的概率分布 [1133]。

在本节中介绍的模型会应用到第 9 章和第 11 章，特别是齐夫法则和 Heaps 法则。

6.5.3 文本相似度

在本节中，我们定义字符串或文档之间的语法相似度的概念。相似度是利用*距离函数*衡量的。例如，对于相同长度的字符串，我们可以定义它们之间的距离为它们拥有不同字符的位置的个数。因此如果它们相同，则距离是 0。这称为 Hamming 距离。距离函数应该是对称的（不受参数顺序影响），并且也应该满足*三角不等式*。即

$$distance(a,c) \leqslant distance(a,b) + distance(b,c) \tag{6-6}$$

一个重要的字符串距离函数是**编辑距离**（Edit distance），或 Levenshtein 距离。编辑距离被定义为使两个字符串相同而在其中任何一个字符串上进行字符插入、删除和替换操作的最少次数。例如 **"color"** 和 **"colour"** 的编辑距离是 1，**"survey"** 和 **"surgery"** 的编辑距离是 2。编辑距离是处理语法错误的首选模型。还有其他更复杂的模型，比如 Soundex 系统，它是一个基于语音的模型 [1195]。编辑距离的概念也存在很多的扩展，比如为插入、删除和替换赋予不同的权重，并且加入移动作为第四种操作。

222 当然也存在很多别的衡量方法。例如，假设我们比较两个给定的字符串且只允许删除字符的操作。在所有的不同字符被删除之后，剩下的字符序列（不一定在原始字符串中连续出现，但是一定要保持相同的顺序）就是两个字符串的*最长公共子序列*（longest common subsequence，LCS）。例如 **"survey"** 和 **"surgery"** 的最长公共子序列是 **"surey"**。

相似度可以扩展到文档中。例如我们可以把文档中的行看做是一个单独的符号，然后计算两个文件行之间的最长公共序列。这个方法用在类 UNIX 系统的 **diff** 命令。这种方法的主要问题是非常耗时，而且没有考虑相似行。后者可以利用行之间的带权重的编辑距离修复。另外一种解决方案包括提取文档的指纹（在某种程度上描述出文档特征的一些文本片段），之后进行比较（如 **siff**），或者找出最大的重复片段。还有一些工具可以提供文档相似度的可视化展示。例如 **Dotplot** 可以绘出一个矩形图，其中两个坐标轴都是文件的行，每个坐标是一个灰度像素，灰度像素依赖于两个对应行的编辑距离。

更有效的文档相似度衡量包括余弦相似度和类似度（resemblance）。在 3.2.6 节中使用文档的向量模型表示和定义余弦相似度，还介绍了其他多个权重函数。需要注意的是非常相似的文档的相似度值接近 1，而差异非常大的文档的相似度值接近于 0。

另外一个相似度衡量是*类似度* [267]。设 $W(d_j)$ 为文档 d_j 中所有不同词的集合，d_i 和 d_j 的类似度函数 $R(d_i, d_j)$ 可定义为：

$$R(d_i,d_j) = \frac{|W(d_i) \cap W(d_j)|}{|W(d_i) \cup W(d_j)|} \tag{6-7}$$

值得注意的是 $0 \leqslant R(d_i, d_j) \leqslant 1$，并且这种衡量方式是 Jaccard 相似度 [816] 的一个特例。而且，我们可以利用任何合理的函数 W 的定义来计算类似度。最高效的技术一直就是片段化（shingling）。片段（shingle）是在文档中连续词的集合，是文档内容的子集。在利用片段来计算 W 时，因为用多个词的组合来代替单个词，所以我们可以更快地计算类似度。

另外，任何在 [0，1] 范围内的相似度衡量都可以通过如下方式方便地转换为距离函数 $D(d_i, d_j)$：

$$D(d_i, d_j) = 1 - Sim(d_i, d_j) \tag{6-8}$$

6.6 文档预处理

文档预处理可以分为五个文本操作（或转换）过程：

1）词汇分析，用来处理数字、连字符、标点符号和字母的大小写。

2）去除禁用词，用来过滤对于检索来说区分度非常低的单词。 223

3）对剩下的词进行词干提取，去除词缀（也就是前缀和后缀），使得文档检索包含查询项的语法变化（例如 connect、connecting、connected 等）。

4）选择索引项或者关键词，这个阶段决定哪些词/词干（或词组）可以用做索引元素。通常一个特定的词能否用做索引项是和这个词的语法特性有关系。实际上，名词通常比形容词、副词和动词携带了更多的语义。

5）创建同义词典等词类结构，或者直接在文本中抽取结构，以便用相关项去扩展原始查询（一个通常有用的步骤）。

接下来会详细讨论这些阶段。在这之前，我们看一下在上面每个阶段完成后得到的文档的逻辑视图。为了方便，图 3-3 在这里被重复用做图 6-6。如同已经讨论的那样，通过聚合这些预处理步骤，我们能够将被系统接受的文档从全文转化到一组更高层次的索引项。

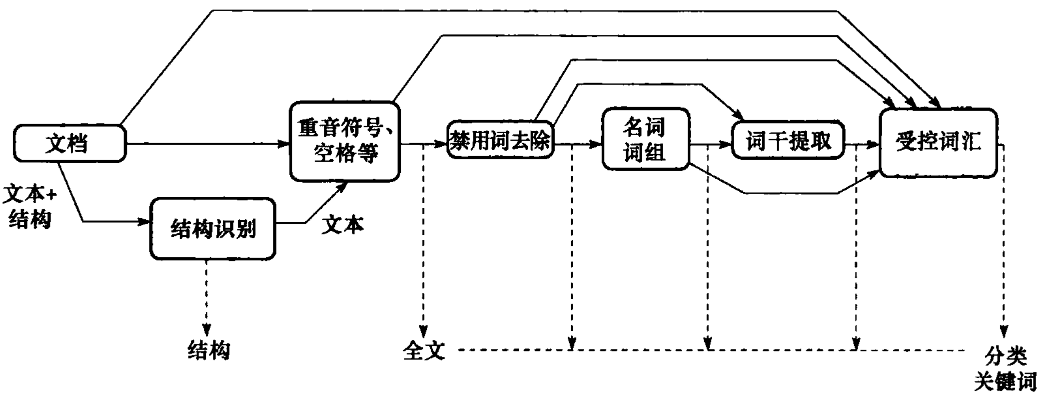


图 6-6　一篇文档通过文本预处理阶段的逻辑视图

6.6.1 文本的词汇分析

词汇分析是将字符流（文档文本）转化为单词流（被选做索引项的候选单词）的过程。因此，词汇分析阶段的主要目的是识别出文本中的单词。乍看，需要做的仅仅是识别出作为单词分隔符的空格（在这种情况下，多个空格被缩减为一个空格）。然而，实际要做的还有很多。例如，下面的 4 种特殊情况也需要仔细地考虑：数字、连字符、分隔符和字母的大小写（小写或大写的情况）。

数字通常不是很好的索引项，因为当没有周围的上下文时，它们的含义模糊不清。例如，假设某个用户对关于在 1910—1989 年因车祸死亡数目的文档感兴趣。这样的一个查询可以由下面的一系列索引项定义｛deaths，car，accidents，years，1910，1989｝。然而，数字 1910 和 1989 在查询中出现可能会导致检索到很多不是关于这两年的文档。这个问题就是因为数字本身太模糊了。因此，通常在索引项中忽略数字是明智的选择。然而，我们也应该考虑那些在一个词中混合出现的数字。例如“510B. C.”（公元前 510 年）就是一个非常重 224

要的索引项。在这种情况下，该使用什么规则就不是很清楚了。而且，一个标识信用卡号码的 16 位数字可能和给定的上下文是高度相关的，在这种情况下，也应该看做是索引项。一个处理数字的初步方法是删除所有包含数字串的单词，除了那些（通过正则表达式）指定的。另外，更深入的词汇分析过程可能会对某些日期和数字进行标准化，以统一格式。这包括了数字和书写版本的日期或数字。

连字符对于词汇分析器来说也是很难判断的。由于用法的不一致，断开含有连字符的词可能是有用的。例如“state-of-the-art”和“state of the art”可以同等看待。然而，有一些包含连字符的词是一个完整的整体，例如 gilt-edge 和 B-49。此外，最合适的过程也是采用一个普遍规则并且指定一些例外情况。

通常，在词汇分析的过程中将完全除去分隔符。虽然，在有些情况下，分隔符是整个词的一部分（例如“510B. C.”），但是删除它们似乎也不会对信息检索产生影响，因为在这种情况下，曲解的几率是很小的。实际上，如果用户在他们的查询中使用了“510B. C”，在查询和文档中同时删除点号不会影响检索。但是，很特殊的情况下可能需要准备一个例外列表。例如，如果一段程序代码出现在文本中，在变量“x. id”和“xid”之间做出区分是比较明智的。在这种情况下，点号不应该删除。

字母的大小写对于索引项的识别来说往往并不是十分重要的。因此，词汇分析器一般将所有的文本转换为小写或者大写。但是，同样在一些非常特殊的情况下，仍然需要做出区分。例如，当用户查找描述类 UNIX 操作系统命令的详细内容的文档时，可能就希望不要做大小写转换，因为大小写也是操作系统协议的一部分。此外，有些语义也可能因为大小写转换而丢失。例如，单词“Bank”和“bank”就有不同的含义——很多其他的词对也存在这样的现象。

Fox[577] 指出，所有这些文本操作能够容易地实现。然而必须要仔细对待，因为它们可能在文档检索的时候产生深远的影响。在用户很难理解索引策略的情况下，就特别麻烦。不幸的是，对于这个问题也没有清晰的解决方案。正如上面提到的，有些 Web 搜索引擎选择不进行文本操作，因为这些操作可以简化用户对检索任务的解释。是否选择这个策略还有
225 待长时间的观察。

6.6.2 去除禁用词

正如在第 3 章中讨论的，在文档集中出现过多的词，它们的区分性往往不好。事实上，在 80%的文档集中都出现的词对于检索目的并没有什么作用。这些词通常称为禁用词(stopword)，并且一般情况下都会被过滤掉而不作为候选的索引项。冠词、介词和连词很显然是禁用词。

去除禁用词还有另外一个重要的好处，它显著减少了索引结构的大小。实际上，只去除禁用词，通常就可以使得索引结构的大小（例如倒排索引大小，见第 9 章）压缩 40%甚至更高。

因为去除禁用词可以为索引结构提供很高的压缩比，因此禁用词表可能会被扩展到其他的词，而不单单是冠词、介词或者连词。例如，有些动词、副词和形容词可能都会被看做禁用词。在文献 [582] 中，给出了一个包含 425 个禁用词的列表。同时也提供了 C 语言版本的词汇分析器。

虽然有这些优点，但是去除禁用词也可能会造成召回率的降低。例如，查询一个包含“to be or not to be”短句的文档。去除禁用词后可能只会留下“be”，从而不可能正确地识别出包括这个短句的文档。这也是为什么有些 Web 搜索引擎采用全文检索（也就是说，在

文档集中出现的所有词都会加入到倒排索引中）的另一个原因。另外，由于 Web 上有很多语言，因此可以通过统计方法定义禁用词，如把出现次数超过一定频率的词当成禁用词。例如，**"html"** 或 **"www"** 等。

6.6.3 词干提取

在很多情况下，用户在查询中使用了一个词，然而在相关的文档中只有这个词的变形出现。复数、动名词和过去分词形式都是这种语法变化的例子，这些变化使得查询单词和对应文档单词不能完全匹配。这个问题可以通过将原单词替换为对应的词干形式来部分地解决。

词干（stem）是在去除了词缀（也就是前缀和后缀）之后的词的一部分。一个词干的典型例子是 connect，connect 是 connected、connecting、connection 和 connections 这些变化的词干。一般认为词干提取对提高检索性能是有用的，因为它们将同一词根的不同变化合并到一个公共概念上。而且，词干提取还有一个减少索引结构大小的作用，因为不同索引项的数量被减少了。

然而，即使支持词干提取的论调从直观上是合理的，但是关于词干提取对检索性能的帮助在文献上仍然存在争论。实际上，不同的研究导致了相当冲突的结论。Frakes[582] 比较了有关词干提取潜在好处的 8 项研究。他倾向于词干提取，但他调查过的这 8 个实验的结果还不能使得我们获得一个满意的结果。因为这些疑问的存在，很多 Web 搜索引擎并不使用词干提取的算法。

Frankes 区分了 4 种类型的词干提取策略：词缀去除、表查找、后续变化和 n 元语法（n-gram）。表查找是在一个表中简单地查找一个词的词干。这是一个简单的过程，但是它依赖于整个语言的词干数据。因为这些数据不能轻易获取，并且还需要相当大的存储空间，这种词干提取算法可能并不实用。后续变化方法基于词素边界的确定，需要使用结构语言学的知识，因此比词缀去除算法更复杂。n-gram 方法基于双字母组合和三字母组合的识别，并且更像词汇聚类，而不是词干提取。词缀去除方法更直观、简单，并且可以很高效地实现。因此本节将只集中讨论词缀去除方法。 226

在词缀去除方法中，最重要的部分就是后缀的去除，因为一个词的大部分变化都是由后缀（而不是前缀）的引入而产生的。已经存在 3、4 个非常有名的后缀去除算法，其中对于英语最流行的是 Porter 算法，因为它既简单又优美。尽管这个算法比较简单，但是与其他更复杂的算法取得的结果差不多。

Porter 算法使用了一个后缀表用于后缀去除。其思想是对于在文本中词的后缀采用一系列的规则。例如，规则

$$s \longrightarrow \phi$$

通过将字母 s 用空（nil）代替将复数形式转化为单数形式。为了识别后缀，我们必须检验在词中的最后一些字母。然后我们寻找能匹配规则集合中规则左边的最长字母串。因此应用下面两个规则

$$\begin{aligned} sses &\longrightarrow ss \\ s &\longrightarrow \phi \end{aligned} \tag{6-9}$$

词"stresses"产生词干"stress"而不是词干"stresse"。通过把这些类似规则分为 5 个不同的阶段，Porter 算法能够提供快速有效的词干提取[⊖]。

⊖ Porter 算法的详细描述可以在本书的网站上找到，其实现发布在 http://snowball.sourceforge.org。

其他语言的词干提取算法可能是非常不同的。例如，西班牙语有很多的例外，因此一个好的词干提取算法也需要字典。像德语和芬兰语这样的粘着语，词干提取就更加困难。对于阿拉伯语也是同样的情况。此外，在汉语中，词干提取也不会起到什么作用。

6.6.4 关键词选择

如果采用了全文表示，那么文本中所有的词都被用做索引项。也就是说，所有的项都需要被索引。另一种方法更具有摘要性，这个方法并不用把所有的词都用做索引项。这就意味着需要选择索引的项集。在目录学领域，选择索引项往往是由专家使用分类体系和受控词汇表（见图 6-6）
227 完成的。在这种情况下，索引项的集合一般比较小，并且每个索引项称为*关键词*（keyword）。

另一个方法是自动选择候选的索引项。可能的做法是识别出名词组（就像在 Inquery 系统［280］中所做的那样），我们现在讨论这个方法。

在自然语言文本中，句子通常是由名词、代词、冠词、动词、形容词、副词以及连词构成的。虽然在每种语法类中的词都以一种特殊目的使用，但可以说基本上大多数的语义是由名词承载的。因此自动选择索引项的一个直观合理的策略就是使用在文本中的名词。可以通过系统地去除动词、形容词、副词、连词、冠词以及代词而实现。

因为两三个名词经常组合成一个单独成分（例如 computer science，计算机科学），所以将文本中相邻出现的多个名词作为一个单独的索引成分（或者概念）是非常合理的。因此我们可以采用名词组来代替单独的名词作为索引项。*名词组*是由一组名词构成的，这些名词在文本中的语法距离（syntactic distance）（利用两个名词之间的词的个数来衡量）不超过一个预定义的阈值（例如阈值为 3）。

当采用名词组作为索引项时，我们根据非基本索引项集合可以获得文档的概念逻辑视图。

6.6.5 同义词典

单词“thesaurus”起源于希腊语和拉丁语，用来指词的宝库［574］。它最简单的形式是由下面两个方面组成的：1）一个预先编辑的、在给定知识领域中非常重要的词的列表；2）对列表中的每个词给出相关词的集合。相关词在它最常见的变化形式中指的是从同义关系中衍生出来的词。

然而，除了一个规范化版本的词汇表之外，同义词典通常进行了词汇表的归一化，还包括了一个比简单词表更复杂的结构以及它们的同义词。例如，最为流行的是由 Peter Roget［1377］创建的同义词典，它包含了短语（phrase），短语是比单词更复杂的概念。Roget 同义词典是一个通用的同义词典（不是为某个特定的知识领域定制的），并且利用类和子类来组织单词和短语。

Roget 同义词典中的一个条目展示如下：

cowardly *adjective*

Ignobly lacking in courage：*cowardly turncoats*.

Syns：chicken（slang），chicken-hearted，craven，dastardly，faint-hearted，gutless，lily-livered，pusillanimous，unmanly，yellow(slang)，yellow-bellied(slang)．

对于形容词 cowardly，Roget 同义词典结合多个同义词组成一个同义词典中的类。Roget 同义词典是通用的，但同义词典也可以为任意知识领域定制。例如，“Thesaurus of Engineering
228 and Scientific Terms”（工程与科学词汇叙词表）就包含了与工程技术相关术语的概念。

根据 Foshkett[574]，同义词典的主要目的基本上是：1）为索引和搜索提供一个标准的

词汇表（或参考系统）；2）帮助用户选择索引项来进行适当的查询描述；3）提供一个分类的层次结构，以便根据用户的要求扩展或者缩小当前的查询范围。然而，在本节中，我们不会讨论如何使用同义词典来修改用户查询。这方面内容在第 5 章中已经涉及了，第 5 章也讨论了如何自动创建同义词典的算法。

创建同义词典的动机是建立在为索引和搜索创建一个词汇表这样一个基本的想法。一个受控词汇表有很多优点，如规范索引概念、减少噪声、识别具有明确语义的索引项，以及基于概念而不是基于词的检索。这些优点在某些特定领域尤其重要，例如对于医药领域中已经有了大量的编辑好的知识。然而对于一般领域，还没有众所周知的文档集知识库。可能是因为文档比较新、太庞大，或者是动态多变的。在 Web 上更是这样。尽管如此，"Yahoo!"搜索引擎为用户提供了一个索引项的层次分类结构，从而减少搜索范围，"Yahoo!"搜索引擎的成功说明，即使是在 Web 的动态世界中，基于同义词典的技术也是非常有用的。

对 Web 来说，对同义词典优点的一致认可还显得太早。因此，很多搜索引擎简单地使用文档中所有的词作为索引项（也就是说，没有为索引和检索目的而使用受控词汇表的概念）。基于同义词典的技术能否在 Web 中兴盛还有待观察。

同义词典的主要部分是它的索引项、项间关系以及为表示这些项间关系的布局设计。索引项和项间关系将在后面介绍。项间关系的布局设计可能是以列表或二维表示的形式出现。这里我们仅仅考虑一般的基于列表的布局结构，因此不会过多地讨论同义词典中项的布局问题。更细节的讨论可以参考 [574]。

1. 关键词

项是同义词典的索引部件。同义词典中的项通常用来表示概念，概念是指传达想法的基本语义单元。项可以是单一的单词、词组或短语，而其中大部分都是单词。而且，项往往是名词，因为名词是最有具体概念的词性。当动词被用做名词时（例如 acting、teaching 等），项也可以是动名词形式的动词。

当一个概念无法用单个单词表达的时候，就用一组词来代替。例如很多概念利用形容词和名词的组合能够更好地表达。一个典型的例子是"polar bear"（北极熊）。在这种情况下，直接索引这个组合词会在 polar 下面而不是 bear 下面生成一个条目，这样显然是不足的。为了防止这个问题的出现，复合项一般被修改成名词作为第一个词。例如我们将"polar bear" 229
修改为"bears，polar"。

注意我们采用了复数形式的"bears"而不是单数形式"bear"。原因是同义词典表示的是事物的类，因此使用复数形式更为自然。然而，对于"body temperature"这样通常以单数形式出现的复合项仍会使用单数形式。决定使用单数形式还是复数形式并不总是件容易的事。

除了项本身外，通常还需要补充同义词典条目的定义或解释。原因是需要在特定的上下文环境中明确词的准确含义。例如，"seal"在"marine animal"（海洋生物）上下文中和在"documents"（文档）上下文中拥有完全不同的含义[⊖]。在这种情况下，要给定义附加上使用的上下文解释，例如"seal(marine animals)"和"seal(documents)"[1501]。

2. 项间关系

给定项的相关项集大多数是由同义词和近义词组成的。此外，项间关系还可以由在文档中的共现次数统计出来。这些关系往往是层次结构的，并且经常指明是广义的（用 BT 表示）或者狭义的（用 NT 表示）相关项。然而，关系也可能呈现横向或者非层次性结构。在

⊖ 分别表示"海豹"和"印章"。——译者注

这种情况下，我们简单地说这些项是相关的（用 RT 表示）。

如第 5 章中讨论的，BT 和 NT 的关系定义了一个分类层次结构，在这个结构中广义项对应一个同义类，而狭义项对应这个类中的一个实例。更进一步说，一个狭义项可能对应两个或多个的广义项（虽然这不是常见的情况）。虽然 BT 和 NT 关系可以全自动地识别（即不需要人的帮助），但处理 RT 关系则更加困难。一个原因是 RT 关系依赖特定的上下文，并且依赖用户组的特殊需求，因此在没有专家提供知识的情况下很难识别。

3. 同义词典在信息检索中的应用

如同 Peter Roget [1377，574] 描述的那样，同义词典是词和短语的分类模式，其组织目的是为了能够促进书面语的思想表达。因此，当用户很难找到合适的项来表示他的想法时（在正式书写时这是常见的情况），他可以使用同义词典来更好地掌握与他想法相关的项的基本语义。

信息检索的研究人员多年来推测和研究了同义词典在进行查询描述过程中的帮助作用。只要用户需要检索文档，首先就要对他想查找的东西进行概念化，这样的概念化是他们的*信息需求*（information need）。给定信息需求，用户仍然必须把它转换为信息检索系统的查询语言。这往往意味着需要选择一系列的索引项。然而，因为这个集合非常庞大并且用户往往
230 缺乏经验，所以这样的初始（initial）索引项的选择可能是错误的或者不恰当的（对于 Web 等未知和高度动态的文档集来说，这种情况更加普遍）。在这样的情况下，重构原始的查询似乎更是一个合理的行为。这样的重构过程往往意味着用相关项扩展原始的查询。因此，使用同义词典来帮助用户查询相关项也是很自然的。

遗憾的是，这个方法并不能一直奏效，因为在同义词典中获取的关系在给定的用户查询的局部上下文中往往是不可行的。另一种选择则是在查询的时候决定同义关系。然而，对于 Web 搜索引擎这个选择也没有吸引力，因为 Web 搜索引擎不能在单一查询中花费太多的时间。这个问题和其他与同义词典技术相关的问题已在第 5 章中讨论过了。

6.7 组织文档

既然组织事务、实体和世界中的物体是人类的特性，那么组织文档集也是很自然的事情。这样，寻找某一类或某一类型的文档就变得比较容易。这是图书馆的起源，否则寻找一个给定的文档会变得非常困难。缺乏组织的文档集使得理解和推理都变得更困难，特别是对大量的文档。现在存在两种主要的组织文档的技术。最古老和最常用的是*分类体系法*（taxonomies），接下来会介绍；第二种是更新的*分众分类法*（folksonomies）。

6.7.1 分类体系法

在众多的组织方式中，最常用的是*层次化*（hierarchical）组织方式。基本原因是因为层次化结构对人们来说更直观。举个例子，在我们整理文件、账单和私人文档时，首先将它们放进文件夹中，然后再放进橱柜中，这是很常见的。另一个比较常见的现象是，在我们整理书桌时，我们将文档和文件叠成独立的堆，使得越重要的堆距离我们越近。公司和机构是层次组织，政府和国家也往往是这样。层次化使得我们可以使用更普遍的概念推理，这样加快了我们的推理，有助于我们对世界的理解。它们也将庞大的（和更复杂的）对象集划分为小的子集以帮助搜索和检索。

对于文档，我们可以把它们分为不同的类。我们利用特化、泛化和同级关系来层次化地组织这些类。用这种方式层次化组织的类形成了一个*分类体系*（taxonomy）。例如在第 8 章

中，图 8-21 的一个例子展示了分类体系。另外一个例子来自本书中，在图 3-2 中我们利用分类体系组织了所有的信息检索模块。

在创建某个特定领域的知识时，分类体系法变得更加有意义。例如在医药领域中，统一医学语言系统（Unified Medical Language System，UMLS）提供了一个关于医学概念的分类学组织 [1174]，这个方法被当做引用标准。可以为不同的领域知识创建分类体系，例如人文艺术、产品与服务、科学技术或者社会科学。怎样自动创建分类体系在 8.7 节中介绍， 231
在本书的其他章节我们也会利用分类体系，特别是在介绍 Web 检索的第 11 章。



图 6-7　利用标签云的 Flickr 分众分类表示（2010 年 1 月）

6.7.2　分众分类法

分类体系法非常有用，但有些时候是不可用的。分类体系法意味用户必须知道一个受控词汇表来描述需要组织的文档或对象，如我们在 6.6.5 节中所见的、在目录系统中的关键词。即使用户知道了这个受控词汇表，这个词汇表中的关键词可能也不能自然地描述文档，并且用户往往希望使用他们自己的词汇作为关键词（参看 6.6.4 节和图 6-6）。在这种情况下的另一种选择是现在所谓的分众分类法（folksonomy）。在分众分类法中，每个用户可以自由地选择关键词，称为标签（tag）。所有标签的集合创建了一个“扁平”的组织结构，文档和对象可以通过标签来查找。当前分众分类法最好的例子是 Del.icio.us[㊀] 中的 URL，Flickr[㊁] 中的图片和 YouTube[㊂] 中的视频。

分众分类法可以和分类体系法结合，并且也有一些研究人员试图自动构建分众分类法的层次结构。最常见的分众分类法的表示是标签云（tag cloud）。在标签云中，流行的标签以字典序展示，并且根据不同的流行度使用不同的字体大小。我们在图 6-7 中给出了一个例 232

㊀ http://delicious.com。
㊁ http://flickr.com。
㊂ http://www.youtube.com。

子，它展示了在 Flickr 中一段时间内最流行的标签。

6.8 文本压缩

文本压缩是用更少的空间来表示文本的技术。压缩算法通过识别和利用文本的规律来创建精减的文本表示。而原始文本可以从压缩的版本中准确无误地重构出来。

文本压缩正在成为信息检索环境中的一个重要问题。数字图书馆、办公自动化系统、文档数据库和 Web 的广泛使用使得在线文本资源爆炸性地增长。在这种情况下，文本压缩成为减少空间开销、输入/输出（I/O）开销和通信时延的很好选择。需要付出的代价是编码和解码文本的时间，但是相比压缩的优点，目前这些代价可以承受。

这里我们主要讨论适用于信息检索环境的文本压缩方法。将文本存储为压缩格式的主要障碍是信息检索系统需要随机读取文本。为了获得在压缩文本中一个给定的词，有些压缩方法需要从头解码整个文本直到获得需要的词。我们将关注那些允许随机访问文本，而不需要从头解压缩的方法。有人可能会提出，大文本可以被切分为很多块，然后每块都单独利用压缩算法进行压缩，从而就可以保证快速地随机访问每个块。然而有效的压缩方法在产生压缩效果前都需要预先处理一些文本。这些块越小则压缩的效果可能会越差。要达到一个可接受的压缩比，最好的情况下需要提前处理 20～50KB 的文本。

在信息检索环境中，采用压缩带来了一些限制，但是也带来了一些机会。例如文本是由单词组成的，这些单词符合某些著名的统计规则，从而利用这些优点可以比大多数通用压缩算法获得更好的压缩比，并且还可以对文本随机访问。

除了空间的节省外，压缩算法还有其他的一些特性需要考虑，例如压缩和解压缩的速度。在很多情况下，解压缩速度比压缩速度要更重要。例如对于文本数据库中的文本只被压缩一次，但是会从硬盘中读出很多次。

压缩方法的另外一个特性是搜索压缩文本而不需要解压缩文本的可能性。在这种情况下，搜索压缩文本可能会更快，因为需要扫描的文本更少。第 9 章给出了如何直接搜索压缩文本的高效方法。

我们首先介绍一些与文本压缩相关的基本概念。之后给出一些对信息检索来说最重要的压缩方法，同时也简要地介绍其他一些相关的压缩技术。接下来，我们讨论主流方法的相对
233 优点。最后，我们会涉及在结构化文本压缩方面的一些最新进展。

6.8.1 基本概念

文本压缩有两个通用方法：*统计方法*（statistical）和*基于字典*（dictionary-based）的方法。*统计方法*估计随后出现的每个文本符号的出现概率。这个估计越准确，就会获得越好的压缩比。*符号*（symbol）可能是字符、文本单词或者固定数量的字符等。在文本中所有可能出现的符号的集合称为*字母表*（alphabet）。估计概率的任务称为*建模*（modeling）。因此*模型*（model）就是下一个符号的概率分布，它依赖于全局的文本统计和在下一个字符之前的符号。使用这些概率，这些符号可以转换为二进制数字，这个过程称为*编码*（coding 或 encoding）。解码器使用同一个模型解释编码器的输出，从而找到原始的符号。两个著名的统计编码方法是霍夫曼（Huffman）编码和算术编码（arithmetic coding）。

字典方法（dictionary method）识别出一系列可以被引用的串（这些串通常称为*短语*（phrase），短语的集合称为*字典*（dictionary））。因此，在文本中出现的短语可以用指向相应字典条目的指针代替，从而进行压缩。字典方法中的建模和编码过程并不存在区别，也没

有短语的明确概率。最为著名的词典方法称为 Ziv-Lempel 系列方法。

预处理成为最近的一个趋势，旨在转换文本以改进压缩比。这类方法中最为著名的是 Burrows-Wheeler 变换，它重新排列文本，使得简单的局部优化方法也能获得很好的压缩。

6.8.2　统计方法

统计方法定义为两个任务的组合：建模任务和编码任务，前者估计每个后续字符的概率，后者把后续符号编码成模型分配给它的概率函数。建模任务往往会给同一个符号一个相同的基于全局文本统计的概率，符号的概率也可以依赖于它前面的几个符号，甚至依赖于目前为止处理过的所有文本。*编码*（code）建立每个符号的表示（*码字*，codeword），而码字是根据概率创建的。克劳德·香农（Claude Shannon）在信源编码理论 [1452] 中建立了模型和编码之间的关系。如在 6.5.1 节中介绍的那样，熵 E 是压缩的下界，每个符号用位衡量，这个结论可以应用到任何基于符号出现的概率分布 $\{p_i\}$ 的编码算法中。压缩中编码的要领是：为了获得更好的压缩比率，越频繁的符号应该赋予越短的码字。

实际上，如果模型给出了一个很偏斜的分布，也就是说一个符号的概率 p_c 比其他符号的更大，那么 $\log_2 \frac{1}{p_c}$会很小。如果编码器赋予符号 c 一个短的码字，则这个短的码字会出现很多次（和 p_c 成比例地），这对压缩是有利的。一个重要的问题是，模型给出的概率 p_c 要接近符号 c 实际的出现概率，否则编码器会给一个并没有真正频繁出现的符号一个短的码字。这也表明，为了获得更好的压缩，1）模型必须正确地估计出现概率；2）编码器必须给 234 符号赋予长度尽可能接近 $\log_2 \frac{1}{p}$ 的码字。

6.8.3　统计方法：建模

压缩模型可以是自适应的（adaptive）、静态的（static）或半静态的（semi-static）；基于字符的或者基于单词的。在本节中，我们会简要地讨论这些模型。

1. 自适应的、静态的和半静态的模型

自适应模型（adaptive model）开始没有包含关于文本信息，并且随着压缩进程的执行而逐步地学习统计分布。当前的模型用于编码每个新的符号，在编码之后这个模型再根据这个新的符号进行更新。自适应模型只需要处理文本一遍，而且除了压缩文本之外也不需要存储其他信息。解压缩器从压缩文本中用同样的方式学习这个分布，在解压缩每个新的符号之前，它已经有了所有的必需信息。对于足够多的文本，这样的模型会收敛于文本的真实概率分布。然而主要缺点是，解压缩文件必须从文件的开始进行，因为关于分布的信息数据是在文件中增量存储的。自适应模型对于通用压缩程序是一个不错的选择，但是对于全文检索来说却不够，因为必须支持对压缩文件的随机访问。

静态模型（static model）对于所有输入的文本假设一个平均分布。对于所有要编码的文本，建模阶段只进行一遍（也就是说，不管事先怎么估计概率分布，之后对于所有要压缩的文本都使用这个分布）。这些模型在数据偏离初始统计假设时会得到比较差的压缩比。例如，对于英文文学足够好的模型，可能在财经文本中表现很差，这是由于财经文本中包括了大量不同的数字，因为每个数字都相对少见，所以会分配长的码字。

半静态模型（semi-static model）并不会假设任何的数据分布，而是在第一遍处理中学习这个分布。在第二遍中，文本会根据第一遍中学到的分布来分配码字进行压缩。关于数据分布的信息必须在转换压缩文本之前传送给解压缩程序。半静态模型的缺点是它们必须处理

文本两遍，并且关于数据分布的信息必须保存下来以被解压缩程序使用。在某些交互数据传输的情况下（例如聊天服务），处理文本两遍可能并不实用。然而，半静态模型却在信息检索中有重要的优点：因为在压缩文件中每一点都使用同一模型，所以直接访问是可能的。

一个简单的半静态模型是基于全局频率信息建立的。令 f_c 是在文本 $T=t_1t_2\cdots t_n$ 中符号 c 的频率（出现的总数）。那么这个半静态模型赋给 c 的概率就是 $p_c=f_c/n$。根据式（6-2），相应的熵是 $E=\sum_{c\in\Sigma}\frac{f_c}{n}\log_2\left(\frac{n}{f_c}\right)$。这个模型可能会也可能不会捕获人们在文本中希望进行压缩的冗余信息。例如，我们使用 2G 的 TREC-3 文档集［704］。在这个简单模型下的每个字
235 符的熵大约是 4.5 位，这就意味着压缩比不能少于 55%。这个压缩比是相当差的，当前最先进的压缩算法可以获得 20%～40%的压缩比。这里的问题并不是没有正确地估计 p_c，而是利用整体频率信息的模型并没有完全捕获英语文本的压缩特性。

正如前面解释的那样，要获得更好的压缩比就要改进模型中的概率估计。在前面的例子中，在估计 c 的概率时我们并没有考虑 c 周围的符号。例如，在英语中，如果我们知道之前的字符是“United Sta”，我们就可以对之后的字符做出更好的猜测。

2. 模型的阶

模型的阶是指用来估计下一个符号的概率而使用的前面符号的个数。在 k 阶模型中，每个符号的概率是由上下文的函数而计算的，这个上下文是由在它前面 k 个符号构成的。最简单的情况是 0 阶模型，在这个模型中，符号概率的计算是与上下文无关的（就像前面的例子）。使用高阶模型使得压缩的效果更好，因为它可能更好地估计下一个符号，但是这个好处并非没有代价。在自适应压缩中，运行高阶模型需要更多的内存。一个很好的例子是*部分匹配预测*（Prediction by Partial Matching，PPM）模型，这个模型可以获得很好的压缩效果。在半静态压缩模型中，为了能够解压缩，较大的模型必须和压缩文件一起存储起来，因此随着 k 的增长会有一个权衡。另外，利用高阶模型的文本压缩必须从开始进行解压，因为任何直接访问都需要知道前面的 k 个符号才能进行解压缩。这点可以通过每个时刻都存储上下文得以缓解。另一个相关技术是将 k 个连续的文本符号看做一个更大的字母表中的单个符号，并在这个新的序列上采用 0 阶模型进行压缩。

继续我们之前在 TREC-3 语料库中的例子，通过使用一个 3 阶模型我们可以减少相应的熵，可以达到 30%的压缩比，但是这个模型必须存储 130 万个频率。为了得到 25%的压缩比，我们需要将模型改为 4 阶，但是需要存储 600 万个频率。因此这里需要权衡的是为了获得高阶的熵，我们也同时需要更大的模型。

3. 基于词的模型

在信息检索中，存在一个更好的解决方案将文本字符分组从而利用相对较小的模型获得更好的压缩比。*基于词的模型*（word-based modeling）使用单词来代替字符作为符号，然后使用 0 阶模型建模单词的序列。通常，一个单词是在集合｛**A**…**Z**，**a**…**z**｝中的连续的字符串，并由不在集合中的字符分割。在信息检索环境中使用基于词的模型有很多的原因。第一，把词当做符号可以获得更好的压缩比，因为词的分布比单个字符的分布更加偏斜：基于词的模型使用 0 阶熵就可以获得 25%左右的压缩比。第二，不同单词的个数相比文本大小不会太大（例如在 2GB 的 TREC-3 语料中有 50 万个单词），因此不需要很大的模型就可以
236 达到和使用较大 k 相同的效果。第三，大多数检索系统是依赖单词这个最小的单元（atom）建立的。为了建立索引，单词已经存储了，因此如果也用在压缩模型里，会进一步减少模型存储的影响。第四，在响应多个词组合的查询中，单词频率也是有用的，因为最好的查询策

略是从频率最低的单词开始（参见 11.5 节）。

因为文本不仅是由单词组成的，而且还包括了分隔符，所以也必须给它们选定模型。处理分隔符有很多不同的方式。因为单词和分隔符往往一个紧随着另外一个，所以会使用两个不同的字母表：一个用于单词，一个用于分隔符。考虑下面的例子“for my rose，a rose is a rose”。在基于词的模型中，词的字母表是 {a，for，is，my，rose}，其频率分别是 2、1、1、1 和 3。分隔符的字典是 {‘,|_|’,|_|}，其频率分别是 1 和 6（|_|代表一个空格）。一旦知道文本是由单词还是分隔符开始的，那么之后使用哪个字母表就没有混淆了。

在自然语言文本中，大多数情况下单词后面跟着一个单独的空格。在 TREC-3 语料的文本中，70%～80%的分隔符都是单个空格。一个更好的模型是把某个单词后面的单个空格看做是这个单词的一部分。也就是说，如果一个单词后面有一个空格，则把这个单词和空格编码为同一个词。否则，需要在编码这个单词之后再编码后面的分隔符。在编码的时候，解码出来的单词会认为后面有一个空格，除非下一个符号也是一个分隔符。现在单词和分隔符（不包括单个空格）使用同一个字母表。例如在前面的例子中，（单个）字母表就是 {‘,|_|’，a，for，is，my，rose}。因为字母表排除了单个空格，因此这个模型称为*无空格单词*（spaceless word）模型。

基于词的模型的良好性质可以用大多数西方语言中的著名统计规则来解释。在 6.5.2 节中有详细的解释，在这里我们将简要地回顾一下。第一个是 Heaps 法则 [730]，此法则说明 n 个词的自然语言文本的词汇表大小 V 的增长是 $V=O(n^{\beta})$，这里 β 是一个依赖于特定文本的常数。对于 TREC-3 文档集 β 取值在 0.4～0.6，这意味着基于词的模型的大小的增长是与 n 的平方根成正比的。第二个是广义齐夫法则 [1794]，该法则说明最常出现的第 i 个单词出现 $O(n/i^{\alpha})$ 次，其中 $\alpha>1$ 是依赖于文本的一个常数。在 TREC-3 文档集中 α 的取值在 1.4～2.0，这因为词汇表的频率是相当偏斜的，同时也解释了为什么基于词的模型有一个非常低的 0 阶熵。

在某些情况下，对于全文数据库，基于词的模型可能会产生大量不同的信源符号，并且必须要小心处理这个问题。例如，在关于词汇分析的一节（在本章的开始）中所讨论的，必须考虑是否把一串数字看做是一个单词。如果是这样，对于一个包含 100 万篇文档的文档集，每篇文档包含文档号作为标识，那么会生成 100 万个由单独数字组成的单词，并且每个数字都只在文档集中出现了一次。这样对于压缩来说是非常低效的。一个可能的解决方案是利用空（或隐含的）分隔符把长的数字分割为较短的数字。这种做法可以减少字母表的大小，从而在压缩比和解码时间上产生可观的改进。

最后需要注意的、很重要的一点是基于词的模型需要长文本才能体现其效率（也就是说，压缩一个单独的网页并通过网络传输出去，这些模型并不能胜任）。当文本比较小的时候（例如小于 1MB），存储词汇表意味着相对大的空间开销。然而对于一般的信息检索任务 237
这并不是问题，因为文本都比较大，而且别的任务，比如检索和查询，也会使用词汇表。信息检索系统有时需要把某些文档通过网络传输给一个不知道全局半静态模型的客户端。在这种情况下，最好是解压缩这个文档，然后利用经典自适应技术来重新压缩它。

6.8.4 统计方法：编码

编码是指根据由模型给定的概率分布而获得符号的表示（码字）。编码器的主要思想是给频繁的符号赋予短的码字，给不频繁符号赋予长的码字，使出现概率为 p 的字符的码字长度尽可能接近 $\log_2 \frac{1}{p}$。正如我们在 6.8.3 节中看到的，概率分布的熵是其码字平均长度

的下界，因此编码器的质量可以由其获得的编码长度与熵的接近程度来衡量。此外，还有一个重要的因素是编码和解码的速度。有些时候，牺牲一些压缩比而减少编码和解码的时间也是很必要的。

在上段中，为了简化，我们假设编码就是*符号编码*。符号编码赋予每个符号一个码字（由整数个位表示），然后连接连续符号的码字而组成编码文件。编码的最低要求是*唯一可解码的*（uniquely decodable）。例如，有三个信源符号A、B和C，并且分配码字如下：A→0、B→1和C→01，那么就没有办法分辨压缩文本"011"代表的是ABB还是CB了。然而编码A→00、B→11和C→110，就是唯一可解码的。但是，为了解码"110000000"，我们必须统计0的总数来决定第一个符号是B还是C。如果每个码字在读到它的最后一位时就可以立即解码，那么这个编码称为*即时码*（instantaneous）。或者说没有任何一个码字是另一个码字的前缀，因此即时码也称为*前缀无关码*（prefix-free code）或者是*前缀码*（prefix code）。

前缀码比较好的可视化方法是二叉树（称为*编码树*）。查看图6-8中的例子。需要编码的符号在叶子结点。每个中间结点有两个子结点分别标记为0和1。对于每个符号，从根到叶子的路径就拼写出了它的码字。

霍夫曼编码（Huffman coding）就是对于给定的概率分布找到最佳前缀编码的方法。也就是，令$\{p_c\}$是一组对于信源符号$c\in\Sigma$的概率。霍夫曼方法为每个符号c赋予了长度为l_c的码字，因此码字的平均长度就是$\sum_{c\in\Sigma} p_c \cdot l_c$，而这个值是所有可能的前缀编码中最小的。另外，霍夫曼编码得到的平均码字长度比由熵编码多出的长度小于1位。这个开销是因为必须赋予每个符号一个整数长度的位。

半静态的基于霍夫曼的压缩方法需要处理文本两遍。在第一遍中，模型决定符号的概率分布并且根据概率分布创建一个编码树。在第二遍中，每个文本符号都根据编码树进行编码。而自适应霍夫曼压缩算法只需要处理文本一遍，其过程中不断增量地更新编码树。输入
238 文本符号的编码也在这一遍文本处理中完成。除了不适用于信息检索系统外，自适应霍夫曼方法也是相当慢的，因为当新的符号读入时更新编码树的开销很大。

*算术编码*是非符号编码的一个例子，也就是说，它不是基于给符号赋予一个整体的码字然后把这些码字连接在一起。在算术编码中，全部输入文本利用一个在0～1的实数的区间来表示。随着输入文本的增多，区间会变得更小，并且用来指定区间的位数也会增多。因为一个出现概率高的符号减少的区间长度要比出现概率低的符号减少的区间要小，因此可以获得压缩。更准确地说，如果模型给出的连续概率是p_1，p_2，…，p_n，那么算术编码会产生一个大小为$P=p_1\times p_2\times\cdots\times p_n$的$[0,1)$中的区间，然后对属于这个区间中的任意一个二进制数编码。因为$\log_2\frac{1}{P}=\log_2\left(\frac{1}{p_1}\times\cdots\times\frac{1}{p_n}\right)=\log_2\frac{1}{p_1}+\cdots+\log_2\frac{1}{p_n}$，所以对概率为$p_i$的每个符号利用$\log_2\frac{1}{p_i}$位进行编码的目标几乎可以完美地实现。因此算术编码器可以获得和文本的熵非常相近的编码。算术编码的更多细节可以参考本章最后的参考文献。

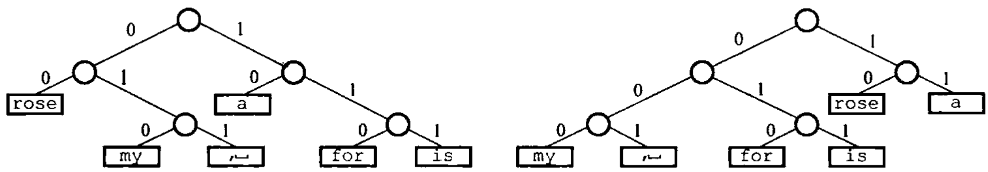
然而在信息检索领域内，算术编码相比霍夫曼编码有几个缺点。第一，算术编码比霍夫曼编码慢很多，特别是使用静态或半静态模型的时候；第二，因为算术编码不是符号编码，所以不能利用算术编码在压缩文件的中间开始解压。而霍夫曼编码不同，如果使用静态或半静态模型时，它可以在压缩文本中任意位置开始的码字进行解码。特别是第二个原因使得算术编码无法在信息检索领域中使用。

综上所述，半静态的、基于词的霍夫曼编码是信息检索系统的一个很好的选择，因为它综合了很多优点：非常好的压缩率、压缩文本的直接访问和很好的压缩与解压缩性能。当需要直接搜索压缩文本时，霍夫曼编码也可以获得很高的效率。然而，最近的密集编码（dense coding）方法提供了霍夫曼编码之外的另外一种选择，密集编码牺牲很小的压缩比，但是却提供了更快的搜索，同时提供了更简单的操作并改进了直接访问。下面将集中讨论基于词的半静态霍夫曼编码和密集编码。

1. 霍夫曼编码

图 6-8a 展示了利用霍夫曼编码压缩单词的一个例子。在这个例子中，单词表中的符号集合是 {‘,|_|’, a, for, is, my, rose}，相应的频率分别是 1、2、1、1、1 和 3（我们假设使用无空格单词模型，因此分隔符‘|_|’并不是单词表的一部分）。

解压缩是按照下面的方式进行的。压缩文件的位流从左到右遍历一遍。在压缩文件中读取的位序列也会从根开始遍历霍夫曼编码树。每当到达一个叶子结点时，对应的单词或者分隔符（组成解压缩符号的一部分）会被打印出来，之后会重新遍历编码树。因此在图 6-8a 中，在压缩文件中的码字‘110’会被解压缩为符号‘for’。 239



原始文本： for my rose, a rose is a rose　　**原始文本：** for my rose, a rose is a rose

压缩文本： 110 010 00 011 10 00 111 10 00　　**压缩文本：** 010 000 10 001 11 10 011 11 10

a)　　b)

图 6-8　a）使用无空格单词的两个可能的霍夫曼树；b）规范（canonical）霍夫曼树

为了创建霍夫曼树，首先需要得到构成字母表的符号，以及它们在需要压缩的文本中的概率分布。然后自底向上地操作，最开始为符号表中的每个符号创建一个结点，这个结点包括这个符号以及对应的概率（或频率）。这时候，形成了一个仅含一个结点的树组成的森林，这些树的概率加起来等于 1。之后，两个带有概率最小的结点被合并，也就是说，它们成为新创建的父结点的两个子结点。这个父结点的概率就是其两个子结点概率的和。重复这样的操作，合并当前森林中概率最小的一对树根，直到仅剩下一棵树，这棵树就是最终的编码树。通过推迟合并拥有高概率的结点对，并把它们放在靠近根结点的位置，从而使得它们的码字更短。每个内部结点的两个分支都标记为 0 和 1。给定 V 个符号和它们在文本中的相应频率，创建霍夫曼树算法的时间复杂度是 $O(V\log V)$。

对于一个给定的概率分布，对应的霍夫曼树并不是唯一的。交换任何一个内部结点的左、右子树都会得到一个拥有相同平均码长的不同编码树。对于大多数应用来说，一般不会使用任意的编码树，它们会选择称为规范树（canonical tree）的编码树，在规范树中任何结点的右子树不能高于它的左子树。图 6-8b 就展示了一个规范树的例子。规范霍夫曼树通过在树的每个层使用信源符号和另外一个数字，可以被很高效地存储。这一点可以在图中很容易地看出来：如果我们按照叶子结点从左到右顺序给定信源符号，以及在每层中叶子结点的个数（也就是从最底层到最顶层，分别是 4、2、0、0），有了这些信息就足够重新构建这棵霍夫曼树。规范霍夫曼树也能够高速解码。在信息检索环境中很需要这些特点。关于规范霍夫曼树的更多细节可以在本章最后的参考文献中找到。

2. 字节霍夫曼编码

霍夫曼提出的原始方法很自然地就导向了一个二进制编码树，因为相比多分支（higher-arity）编码树它们可以获得最好的压缩比。然而，也可以对每个符号都赋予整个字节（byte）的序列。这样，霍夫曼树的度数就不是 2 而是 256。

如果考虑一个基于词的模型，使用字节代替位进行编码，会使得压缩比降低到 30%左 240 右，但是这种方法仍然具有竞争力，就像我们在后面将要介绍的一样。作为交换，字节霍夫曼编码的解压缩速度比二进制霍夫曼编码更快，因为它不需要位的移位和掩码操作。

字节霍夫曼编码的另一个优点是在压缩文本中直接搜索会变得更简单，因为可以像标准字符串匹配那样基于字节地进行操作。在压缩文本中搜索某个单词的时候，我们首先要在词汇表中找到它。这可以通过序列扫描获得（利用在第 9 章中介绍的任意一种技术）或者创建一个可以加速搜索的数据结构。注意能够接受序列扫描的原因是词汇表的增长是比较慢的。对于 2G 的 TREC-3 文档集，仅仅需要 5M 的词汇表，在 1/10 秒内就可以扫描完成（词汇表一般是放在内存中的）。一旦我们在词汇表找到这个词后，我们就会在霍夫曼树中标记相应的叶子结点。然后，如同解压缩时一样操作文本，只是我们到达叶子结点不需要输出符号，而是报告要搜索的单词出现。这个过程是很简单的，仅仅扫描压缩文本中所有的字节一次，并且在扫描每个字节的时候仅有非常少量的工作。因此，搜索的速度是非常快的。特别是，它比在原始文本中进行序列扫描要好得多，因为只需要原来 30%的 I/O 操作。需要注意的是，对于位霍夫曼编码也可以使用相同的方法，但是位操作使得这个方法会慢一些（如果一次要立即处理多个位时会变得更加复杂）。

这个方法可以很好地扩展到搜索一组词的集合。例如，搜索一个单词的所有变体或者搜索一个扩展的查询。这种情况下，我们用多模式搜索算法（参看第 9 章）扫描词汇表，然后标记我们需要报告的所有单词。一旦完成这些工作，扫描压缩文本的过程就一样了。

最后，假设我们希望搜索更复杂的模式，例如通配符、字符的范围，甚至正则表达式。而且，假设我们希望在查找结果中允许一定数量的错误。所有这些模型都有高效的序列搜索算法，将在第 9 章中介绍。我们仅仅需要在词汇表上利用这些序列算法，然后每当在整个词汇表中找到一个匹配，我们就会标注这个单词。注意这些工作只是在词汇表上进行，相比整个文本的工作量是很少的。一旦标记好相关的词，我们只需要在压缩文本中执行简单的字节扫描算法。

搜索的所有复杂性都包含在词汇表扫描中。越复杂的模式序列，对应的搜索算法就越慢，但是这些算法只应用在整个文本中的很小一部分（词汇表）。搜索的主要部分对于复杂度并不敏感。因此，当涉及复杂模式时搜索压缩文本比搜索原始文本要快 8 倍，而涉及搜索简单模式时快 3 倍（因为减少了 I/O 开销）。

我们之前仅考虑了单个词的查询（或者匹配一个词的复杂模式）。之前提到的方案可以很好地扩展到匹配短语和一般的模式串，但是这些技术需要更加复杂的模式匹配概念，在第 9 章中我们会进行讨论。在本章中我们仅仅考虑简单的情况。

尽管这个搜索技术非常简单并且具有一致性，但它对于单个词的查询（一般来说，简单查询比复杂查询更频繁）表现得更好。我们可以在词汇表中找到单词，获得它的码字，然后 241 使用在 9.5 节中的任意字符串匹配算法在压缩文本中查找码字。这样就可以使用不检查就跳过某些字节的更快的搜索技术。例如，Horspool 字符串匹配算法（参见 9.5.1 节）在这个情况下可以快 3 倍。

不幸的是，这个算法不能适用于霍夫曼编码。因为有可能两个码字的结合包括第三个码

字，这种情况会错误地识别为搜索命中。例如在二进制编码中，有如下编码：A→0、B→10、C→110 和 D→111。那么 DB 可以编码为“11110”。如果我们查找 C，我们会把它转换为搜索“110”，当组合 D 和 B 的码字时，我们会错误地报告一次搜索命中。而且，除非我们从开始重新扫描文本，否则我们无法检验出搜索命中是否是误报的。

下面介绍密集编码，它是字节霍夫曼编码之外的另一个选择，可以解决上面的问题，而且也能在其他复杂搜索中使用。实际上，密集编码在各个方面都优于字节霍夫曼编码。

3. 密集编码

密集编码比字节霍夫曼编码更简单，和基于词的模型结合时，产生与之相当的压缩表现。它也是给高频符号以更短的码字，然而这里的编码只依赖于符号的*排名*（rank），也就是按频率的降序排列后的位置，而不是依赖于它的实际频率。密集编码可以看做是整数变长编码，这也是一种常见的压缩解决方案。然而，当用它来压缩自然语言文本时，密集编码经过了仔细地研究，重点在压缩文本中快速地直接搜索。

一个简单的密集编码如下所示。将符号按照频率的降序排列，然后根据它们的位置赋予它们之后的数字：第 i 个最常见的符号赋予数字 $i-1$。这个数字写成一个变长的字节序列，其中每个字节中的 7 位用来编码数字，最高的 1 位保留下来作为每个码字的最后字节的标示。

更准确地说，排序在 1～128 的符号的码字是 0～127，并且它们都获得一个单字节码字。这个字节也是它们码字的最后一个字节，因此它们最高一位设置为 1（也就是加上 128）。我们把最高位设置为 1 的字节称为*停止符*（stopper），因为它们表示码字的结束。其他的字节称为*持续符*（continuer）。因此排位为 1 的符号获得码字〈128〉=〈0＋128〉，排在第 2 的符号获得码字〈129〉=〈1＋128〉，按照上面的规律，直到排序为 128 的符号获得码字〈255〉。排序从 128＋1＝129 到 $128+128^2$＝16 512 的符号被赋予从〈0，128〉到〈127，255〉的二字节码字。注意这个方法相对于单纯的基于 128 数字有点奇怪，因为数字 0 出现在数字前。第一个 3 字节的码字〈0，0，128〉被赋予第 16 513 个符号，并且直到第 $128+128^2+128^3$＝2 113 664 个符号，它被赋予码字〈127，127，255〉。很少有文本词汇表能大到需要 4 字节的码字。表 6-1 给出了利用密集编码的示例。 242

表 6-1　密集编码

词序号	码字	字节	词数
1	〈128〉	1	
2	〈129〉	1	
…	…		128
127	〈254〉	1	
128	〈255〉	1	
129	〈0，128〉	2	
130	〈0，129〉	2	
…	…		
256	〈0，255〉	2	
257	〈1，128〉	2	128^2
258	〈1，129〉	2	
…	…		
16 511	〈127，254〉	2	
16 512	〈127，255〉	2	
16 513	〈0，0，128〉	3	
…	…		128^3
2 113 664	〈127，127，255〉	3	

需要注意的是，因为最高位作为一个码字结束的信号，因此这个编码自然就是一个前缀编码：一个码字不可能是另外一个码字的前缀。根据它们的最高位，使得较短码字的停止符字节不可能和较长码字的持续符字节相同。因此，密集编码通过一个简单的方式获得了霍夫曼编码的两个重要性质（是前缀编码并且偏爱更加常见的符号），但这个编码显然不是最优的。

密集编码有一个很好的性质，使得它们成为信息检索领域中最受关注的编码：它们是自

同步的（self-synchronizing）。这意味着，给定压缩文本中的任何一个位置，它都非常容易地判断出下一个或者前一个码字的开始，这是由于字节最高位的缘故。而对于一般的霍夫曼编码这是不可能的。例如，对于（二进制）霍夫曼编码的码字“10”、“01”、“11”、“000”和“001”。从“11010101010101”的中间解码文本是不可能的，我们必须从文本的开始去观察如何开始编码。

我们一直坚持认为霍夫曼编码可以随机访问。确实，用霍夫曼编码的文本可以从任意一个码字的开始位置进行解码，而不是从任意位置。密集编码在这个意义上更加强大，因为它可以从任意位置开始解码，不管它是不是一个码字的开始。这只是寻找之前或之后的一个停止符的问题，之后就可以从那里开始进行解码了。特别地，给定任意一个位置，通过反复定位之前的一个停止符并且每次解码一个码字，可以在这个位置上进行反向解码。对于霍夫曼编码来说，即使给定了码字的开始，这也是不可能的。这个特点非常有趣，例如，显示压缩
243 文档中感兴趣的一个位置周围的片段。

这个同步性质使得密集编码可以实现字节霍夫曼编码无法实现的更快速的查找。为了查找一个词，我们需要获得它的码字，然后就可以使用任意的字符串匹配算法在压缩文件中搜索了。根据最高位，很容易区分虚假的搜索命中和真实的搜索命中：查询码字的最后停止符必须匹配文档中的停止符（码字的结束）。虚假命中只有在一个码字是另一个码字的后缀时才发生。也就是说，假设我们查询码字“a b $\overline{\text{c}}$”，我们在这个码字的停止符字节上画了线。如果在编码中存在码字“d a b $\overline{\text{c}}$”，那么我们可能在文本“…e f $\overline{\text{g}}$ d a b $\overline{\text{c}}$…”中找到我们的码字。然而只要查看一下之前的文本，“d”并不是一个停止符，因此我们匹配了一个不同的码字。这样一个快速简单的检查在霍夫曼编码中是不可能实现的。

通过合并码字然后搜索这个组合，使得搜索短语也是可能的（单个字符串匹配算法会随着搜索字符串的增长而更快，参见第 9 章）。搜索复杂模式可以用下面两种方法。一是通过使用同样的方法（标记词汇表单词，然后按字节搜索文本），或者通过收集所有感兴趣的单词的码字，然后对它们同时进行多模式匹配。随着模式数量的增长，这种类型的搜索会变差（见第 9 章），因此当搜索词数量适中时，它比简单的基于字节搜索的方法可能好也可能差。

(s，c）密集编码

在不损失任何性能的前提下，改进密集编码的压缩表现也是有可能的。这个编码将 0～127 的值分配给持续符（continuer）字节，将 128～255 的值分配给停止符字节。这个划分是任意的，并且可以修改为 c 个持续符和 s 个终止符，满足 $c+s=256$。通过考虑字节的数值，停止符和持续符仍是可以区分开来的。现在，前 s 个最常出现的符号用一个字节编码，后面的 sc 个用 2 个字节编码，再后面的 sc^2 个用 3 个字节编码，以此类推。在 TREC 文档集的实验中，s 的最优值为 185～200。利用这些最优值，密集编码的压缩率非常接近于字节霍夫曼编码（仅差了 0.5%～1%）。

寻找最优 s 和 c 的值代价并不高。假设 V 个符号已经根据其频率 f_i 倒序排列。我们首先在第一遍中计算累计频率，$F_0=0$ 和 $F_i=F_{i-1}+f_i$。然后用给定的（s，c）组合进行压缩的文件长度是：

$$1 \times F_s + 2 \times (F_{s+sc} - F_s) + 3 \times (F_{s+sc+sc^2} - F_{s+sc}) + \cdots$$

上面的和式仅有 $O(\log_c V)$ 项，并且也很容易计算出所有 256 种可能的（s，c）组合来得到最短压缩文件的组合。

在本章的开始我们提到，在信息检索系统中使用压缩的唯一缺点可能是实现复杂度。在

这方面，密集编码也有优势，因为它们比处理霍夫曼编码更简单。例如，不需要创建也不需要存储编码树；仅仅存储词汇表就足够了。一个简单的代数运算就可以将词的排序转换为编码，反之亦然。 244

总之，密集编码可以获得几乎与字节霍夫曼编码相同的压缩率，并且也有霍夫曼编码所有的有利特征。然而，密集编码的编程和操作更加简单和快捷，有很强的同步属性，并且可以快速搜索。位霍夫曼编码仍然是一个不错的选择，因为它们可以获得更好的压缩比。

6.8.5　字典方法

字典方法通过把一组连续的符号（或短语）用一个指向字典条目的指针代替，从而获得压缩效果。因此在字典方法的设计中，最主要的问题是字典条目的选择。短语的选择可以通过静态、半适应（semi-adaptive）或自适应算法完成。最简单的字典方案是使用包括短词组的静态字典。静态字典编码非常快，因为它们几乎不费力气就可以获得少量的压缩。一个以多种形式多次提到的例子是双字母组合编码（digram coding），这种方法选择一对词，然后利用特殊的字符代替。静态字典很少能够非常有效，因为适合一篇文本的字典可能对另一篇文本并不适用。

自适应字典方法更加成功。这种类型算法最出名的技术是 Ziv-Lempel 系列的压缩算法。在 Ziv-Lempel 压缩算法中，字典随着文本压缩的过程创建。插入到字典中的字符串属于已经处理过的文本。根据不同的 Ziv-Lempel 压缩算法，会引用不同的子串。在 LZ77 压缩算法中（这个算法在很多广为流行的压缩软件中得到了实现，例如 **zip**、**pkzip**、**gzip**、**winzip**、**arj** 等），已经处理过的任意文本都可以称为候选短语，因此可以被引用。另外一些变化允许产生更少的短语，这样既可以加速压缩过程，还可以减少指向字典的指针的大小。

因为它们的速度、内存的节省以及较高的压缩比，Ziv-Lempel 算法作为通用压缩算法非常流行。然而，它们也与在信息检索领域中使用的其他自适应统计模型具有相同的缺点：在随机位置上访问压缩文本变得非常困难。在信息检索环境中，半静态的、基于词的统计方法在各个方面都优于 Ziv-Lempel 方法。

对于信息检索来说，更有吸引力的是 Re-Pair 压缩方法，这是一个半静态字典技术，它先找到最常出现的一对符号，然后用一个新的符号来替代它们。Re-Pair 方法的第一步是为字母表中的每个符号赋予一个整数，然后将文本重写为相应的整数序列。然后迭代进行，最频繁出现的连续的数字对 A·B 首先被识别出来。规则 C→A·B 将一个新的整数 C 插入到字典中，然后所有 A·B 的出现都会被 C 来代替（C 称为短语（phrase））。这个过程一直重复，直到没有连续对在文本中重复出现。注意像 C 这样的短语也可能根据规则 E→D·C 参与到新的短语中。因此这个字典可以看做是短语的二叉层次结构，其中叶子是字母表中的符号，中间结点是根据其他短语或符号建立的。一旦获得最后的序列，可以停止压缩，也可以进行下一步的 0 阶压缩。 245

Re-Pair 压缩方法能够获得非常好的压缩比，并且它可以简单地在任意位置开始解压缩文本：它需要的只是序列的短语标识，然后利用字典输出对应的符号序列。它的主要缺点是压缩非常慢，并且需要大量的内存。文本集必须分块压缩，使得每块都可以在内存中处理。

然而，有一种 Re-Pair 方法对于信息检索应用非常有吸引力。如果我们将文本看做是词的序列，并运用和整数序列压缩同样的算法，那么虽然压缩率会差一些，但是在字典中的所有的短语都是词的序列。更准确地说，它们是文本中的频繁短语。这个字典可以应用于短语

浏览（phrase browsing），从而使得用户可以在文本集的字典中浏览从而找出相关的短语。例如，用户可能写下如“United”这样一个词，然后查看在文本所有出现在它后面的多于一次的词。利用这个方法，有可能识别出在文本中感兴趣的短语是“United Nations”、“United States”和“Manchester United”。例如，如果用户选择了第一个短语，那么他会知道还存在像“United Nations Development Program”和“United Nations Environment Program”等相关短语。完成以上任务不需要检查文本，而仅仅需要查看字典就够了（见第 5 章中的查询扩展技术）。

最后，在压缩文本中搜索单词也是可行的。它需要做的只是找到所有包含这个单词的词典短语，然后同时搜索所有这些短语（参见第 9 章）。尽管搜索短语和更复杂的模式有些复杂，但是仍然是可行的。

6.8.6 压缩预处理

文本压缩的最新趋势是压缩的预处理过程。其思想是将文本转换为更容易压缩的形式。这个方法最成功的代表可能是 Burrows-Wheeler 变换（Burrows-Wheeler transform，BWT）。

BWT 是对文本位置的重新排列，因此它本身并不压缩。然而，在转换后的文本上用简单的局部优化就可以获得原始文本的高阶压缩。

一个文本 $T = t_1 t_2 \cdots t_n$ 的 BWT 如下定义。假设 $t_n = \$$ 是一个特殊的终结符，比其他的都小。构造一个概念矩阵，矩阵的行是文本 T 所有的*循环移位*（cyclic shift），即对于所有的 i 有 $t_i t_{i+1} \cdots t_n t_1 t_2 \cdots t_{i-1}$ 。按照字典序排列 M 的全部行。令 F 为 M 的第一列，L 是最后一列。则 T 的 BWT 就是 L。在图 6-9 的例子中，$T=$“mississippi$”的 BWT 是 $L=$“ipssm$pissii”。

很惊奇的一点是，可以从 L 中将 T 恢复出来，达到这个目标的主要任务称为 LF 映射（LF-mapping）。给定在 L 中的一个位置 i，LF 映射可以找到在 F 中的 $L[i]$ 的位置。例如，如果 $L[5]=$“m”。很容易看出它肯定是 $F[6]$（因此 $LF(5)=6$），因为 F 以字典序包含了 T 中所有的字符，并且在 T 中有 5 个比“m”小的字符。这就能够存储数组 C，在 C 中存储的信息是，对每个字符 c，在 T 中小于 c 的字符的个数。对于 $LF(4)$ 的情况更复杂一些，因为 $L[4]=$“s”，而在 F 中有 4 个 s。到底是哪个对应 $L[4]$ 呢？BWT 的一个很好的性质是 L 中 c 的第 i 次出现，对应 c 在 F
246 中的第 i 次出现，因为在两种情况下它们都根据在 T 中处于 c 后面的文本进行排序。于是，因为 $L[4]$ 是 L 中的第二个“s”，它对应于在 F 中的第二个“s”，也就是说 $F[10]=F[C[\text{“s”}]+2]$，因此 $LF(4)=10$。为了能够高效地计算 LF 映射，还需要一个

m	i	s	s	i	s	s	i	p	p	i	$
i	s	s	i	s	s	i	p	p	i	$	m
s	s	i	s	s	i	p	p	i	$	m	i
s	i	s	s	i	p	p	i	$	m	i	s
i	s	s	i	p	p	i	$	m	i	s	s
s	s	i	p	p	i	$	m	i	s	s	i
s	i	p	p	i	$	m	i	s	s	i	s
i	p	p	i	$	m	i	s	s	i	s	s
p	p	i	$	m	i	s	s	i	s	s	i
p	i	$	m	i	s	s	i	s	s	i	p
i	$	m	i	s	s	i	s	s	i	p	p
$	m	i	s	s	i	s	s	i	p	p	i

↓

F											L
$	m	i	s	s	i	s	s	i	p	p	i
i	$	m	i	s	s	i	s	s	i	p	p
i	p	p	i	$	m	i	s	s	i	s	s
i	s	s	i	p	p	i	$	m	i	s	s
i	s	s	i	s	s	i	p	p	i	$	m
m	i	s	s	i	s	s	i	p	p	i	$
p	i	$	m	i	s	s	i	s	s	i	p
p	p	i	$	m	i	s	s	i	s	s	i
s	i	p	p	i	$	m	i	s	s	i	s
s	i	s	s	i	p	p	i	$	m	i	s
s	s	i	p	p	i	$	m	i	s	s	i
s	s	i	s	s	i	p	p	i	$	m	i

图 6-9 创建“mississippi$”的 Burrows-Wheeler 变换

$Occ[i]$ 数组，这个数组告诉我们 $L[i]$ 在 $L[1, i]$ 出现的次数。更准确地说，$LF(i) = C[L[i] + Occ[i]]$。

知道了 LF 映射，我们现在就可从 L 中恢复 T 了。关键的属性是 $L[i]$ 在 T 中总是在 $F[i]$ 的前面。因为我们知道 $F[1]$ 必须是 \$（因为它是在 T 中最小的字符），并且这是 t_n，于是有 $t_{n-1} = L[1]$。之后我们必须知道 $L[1]$ 在 F 中的位置。这个位置是 $i = LF(1)$。因为 $F[i] = t_{n-1}$，于是有 $t_{n-2} = L[i]$。现在我们移动到 $i' = LF(i)$ 的位置，然后找出 $t_{n-3} = L[i']$，并且以次递推，直到以逆序恢复整个文本 T。

现在我们给出一些为什么 BWT 可以有助于压缩的直观感觉。例如某些常见的短语，“United States”。所有的在 T 中出现的“ed States”会在 M 中连续出现。在大多数情况下“ed States”在“t”的后面出现，因此 L 将在连续的区域包含大多数“t”。类似地，“ted States”、“ited States”和“nited Stages”在 L 中会分别产生长程的字母“i”、“n”和“U”。这说明 BWT 的关键点是：在 T 中的第 k 阶的冗余转换为在 L 中的不同几个字符的持续出现。 247

很显然，通过编码这些持续出现的相同字母并且对于 L 使用一些简单的优化，对于一个适当的 k，压缩可以达到 T 的一个 k 阶熵。

一个广泛应用使用的 BWT 压缩软件是 bzip2，目前在大多数 UNIX/Linux 发布版中很常见。实际上，BWT 压缩软件并不显式地创建 M 矩阵，而是利用后缀数组直接构造 L（参见 9.4 节）。L 中的字母是后缀之前以词典序排列的那些字母。

6.8.7　文本压缩技术的比较

在本节中，我们从概念上对之前提到的最有前途的压缩算法进行比较，主要考虑那些与信息检索最相关的方法和在市场上最广泛应用的方法。在统计模型中，我们选择了两个半静态模型和一个自适应模型。半静态模型使用了基于词的 0 阶模型和两个不同的编码器：位霍夫曼和字节密集算法。自适应模型采用算术编码的 PPM 建模（我们使用 **ppmdi** 程序作为比较的代表）。我们还选择了字典方法中的两个代表，半静态的例子是 Re-Pair，而 Ziv-Lempel 系列中的 LZ77 的一个变种作为动态的代表（例如 **zip** 或者 **gzip** 程序）。最后我们选择了 BWT 变换来代表基于文本预处理的方法（更准确地说，我们使用了 **bzip2** 程序）。

表 6-2 从多个因素比较了之前考虑的几种方法：压缩率、压缩速度、解压缩速度、内存空间开销、（单词）直接搜索能力和随机访问能力。

表 6-2　主要压缩技术的比较

测度	Huffman	Dense	PPM	Re-Pair	LZ77	BWT
压缩率（%）	25～30	30～35	20～25	20～30	30～40	25～30
压缩速度	快	快	慢	非常慢	快	慢
解压速度	快	非常快	慢	快	非常快	慢
内存空间	高	高	高	高	低	高
直接搜索	快	非常快	否	快	慢	否
随机访问	是	是	否	是	否	否

任何压缩方法最重要的目标之一是获得更好的压缩率。表 6-2 展示了在英文文本中获得的典型压缩。压缩方法的另外两个重要特征是压缩和解压缩的速度。然而衡量各种压缩方法的速度是比较困难的，因为它依赖于算法的实现和用来运行程序的机器的体系架构等。我们考虑更粗糙的类别，例如慢、快等。类似地，对于内存使用，我们也只是考虑高和低两个类

别（1M左右就认为是高）。直接搜索也分为粗糙的速度类别，然而有些方法根本不支持直接搜索。对于随机访问，我们仅仅是指它能否在指定位置开始解压缩，而不需要解压缩文件中的许多部分。

统计（基于词）的方法，即使是面向字节的，也能获得比 Ziv-Lempel 方法更好的压缩
248 比。特别是基于词的密集编码对于足够大的文本集（大于数 M）甚至所有方面都比 Ziv-Lempel 表现得更好。PPM 和 BWT 获得了比密集编码更好的压缩效果。原因是这两种方法利用不同的方式获得了高阶模型，并把它和算术编码结合起来。然而代价是，它们的压缩和解压缩速度都很慢。基于词的位霍夫曼编码用更好的时间表现获得了相同的压缩比。另一个比较慢的压缩方法是 Re-Pair，但是它基于字符（character-based）的变种获得了很好的压缩比。基于字的变种获得了接近30%的压缩比，但是这个方法可以允许直接搜索单词。所有其他方法在时间上表现很好，但是密集编码和 Ziv-Lempel 在解压缩时间上表现更加突出，因为它们更简单。除了 Ziv-Lempel 方法外，其他所有方法都需要相对较高的内存空间以获得可接受的压缩，Ziv-Lempel 仅利用 64KB 内存就可以很好地运行。除了 Re-Pair 和一些 PPM 的变种外，内存需求对于现代计算机来说完全不是问题。对于直接搜索，可以看出密集编码是最有前途的选择。最后，只有半静态方法实现了对于信息检索系统非常关键的对压缩文本的随机访问。另外 Re-Pair 是唯一一个允许短语浏览的方法。

6.8.8 结构化文本压缩

结构化文本是表示信息的普遍方式。XML 作为一个存储、交换和操作半结构数据的格式被广泛使用，也使得结构化文本在很多信息检索系统中成为表示文档的常用格式。这个结构以很多方式使用，特别是用于改进检索（见第13章）。另外，它也可以利用这个结构获得更好的压缩效果。

经典的文本压缩软件并没有将结构信息考虑在内，这样会失去获得更好压缩效果的机会。直到最近，才考虑利用结构信息改善压缩性能。因为还没有为结构化文本压缩建立理论，所以我们选择性地提及最突出的产品和原型（关注与信息检索领域最相关的那些），及其主要思想。

1. XMLPPM

从压缩率的角度看，这可能是最好的结构化文本压缩方法。然而，XMLPPM 是自适应的，因此不能支持压缩文件的直接访问。XMLPPM 依赖于类似 PPM 建模和算术编码。同时也可以选用其他几个模型，在压缩的进程中，XMLPPM 可以在它们之间进行切换。相反，在整个过程中只能使用一种算术编码。使用的4个模型是：一个用于元素（element）和属性名，一个用于元素结构，一个用于属性，还有一个用于字符串。对于某些模型，需要给定 PPM 模型树的层次结构：当一个新的结构元素开始时，将到达结点的树路径的标签序列用做上下文，而不是利用之前的字符（来自另外一个树结点）。例如，如果信源文本是“<author><title>Prof. </title>Claude Shannon</author>”，那么用来建模“prof.”
249 的2阶 PPM 上下文就是“<author><title>”（就像它们被用做信源符号），用来建模“Claude”的上下文是“<author>”，而用来建模“Shannon”的上下文是“<author>Claude”。

2. SCM：结构化上下文建模

SCM 可以看做是 XMLPPM 的一个简化，它支持直接访问。SCM 仍然基于这样一个直觉，即在一个给定标签下的文本应该有相似的统计结构，而不同于在其他标签下的文本。所

提出的方法是将每个标签下文本的建模和编码与其他标签中的分隔开来。例如，在一个电子邮件（E-mail）集合中，人们可以认为像〈from〉和〈to〉这样的字段会包含电子邮件地址，〈date〉字段包含日期，并且〈subject〉和〈body〉包括自由文本。SCM 的提出者对每个标签使用基于词的位霍夫曼编码器来说明这个想法。他们提出了一个方法，当分布被证明是相似的，这个方法会合并标签，以防止重复的词汇。在 TREC-3 中，SCM 比一般的基于词的位霍夫曼编码获得了 2%～4%的压缩改善，并且保持了直接访问和搜索的能力。

3. LZCS：压缩结构的 Lempel-Ziv 方法

LZCS 的目标是那些有很多重复的结构化集合，例如 Web 表单库，如订单和发货单这样的电子商务文档，Web 服务交换文档等。这个方法将结构中任意子树用一个指向之前见过的相同的子树来代替。除了限制在完整的子树上外，另一个与普通的 LZ77 压缩算法的主要不同点是这些指针是指向压缩文件的位置，而不是未压缩的文件。因此，可以很容易地在压缩形式中浏览文档，而完全不需要将它解压缩。此外，结果文档仍然是一个可用的 XML 文档，仍然可以没有任何限制地传输、可视化和操作。利用经典的压缩算法再进行一遍压缩，往往可以比直接在原始文件中压缩取得更好的压缩效果。

6.9 趋势和研究问题

目前，仍出现很多改变或新的提议，并且非常迅速，特别是因为 Web 的出现。在这点上，读者一定被一堆的缩写弄糊涂了（我们也是这样），尽管事实上，我们只是提到了最相关的语言和格式。它们当中最重要的包括在本书的词汇表中。有些人认为像 CSS 或 XML 这样的新的格式带走了 HTML 的简洁，而这正是 HTML 成功的基础。未来将会告诉我们哪种观点会取得胜利。图 6-10 展示了涉及的主要语言的分类体系。实线代表一种元语言的实例（例如 HTML 是 SGML 的实例），而虚线表示衍生出来的语言。不同成果的收敛和聚合是主要的趋势，而 Web 正在成为主要的应用。

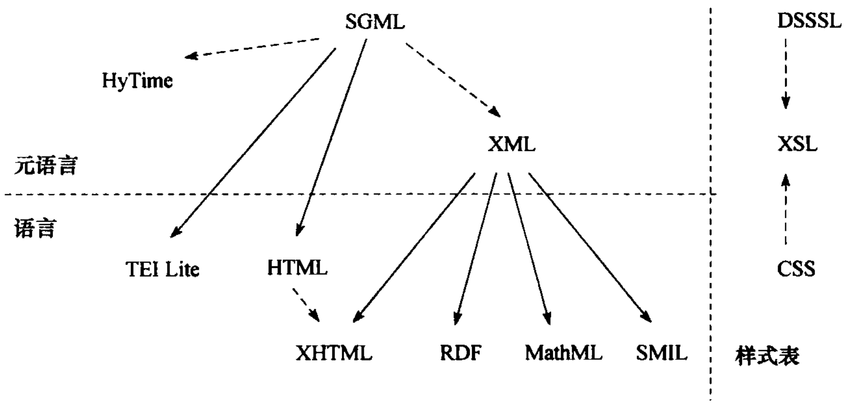


图 6-10　文档语言的分类体系

在欧洲，替代 SGML 的是开放文档体系结构（Open Document Architecture，ODA），这也是一个标准（ISO 8613）。ODA 用于共享电子文档，而且不失去对内容、结构和文档布局的控制。ODA 定义了一个逻辑结构（像 SGML 一样），一个布局（layout）和内容（包括 250
向量和光栅图像）。ODA 文件可以是格式化的（Formatted）、可处理的（Processable）或者格式化可处理的（Formatted Processable）。格式化文件包括关于内容和布局的信息，不能被编辑。另外两种类型是可以被编辑的。可处理文件除了内容也包括了逻辑信息，而格式化

可处理文件包括全部。ODA 现在并没有被广泛使用（见第 14 章）。

最近的发展包括：

- XForms 的发展，这是一个用来表示下一代 Web 表单的 XML 应用。其主要想法是将传统的 XHML 表单分为 3 个部分：XForms 模型、数据实例和用户接口。这个方法将内容和表示区分开来，允许重用，提供了强类型，因此减少了服务器调用的次数，同时提供了设备独立性并减少了对脚本的要求。XForms 并不是单独使用，而是整合到如 XHTML 或者 SVG 等其他标记语言中。
- VRML 和动态 HTML 的整合，为 HTML 和 Web 浏览器提供了一系列发展的特征和体系扩展，包括串联式样式表和文档对象模型。
- 产品模型数据交换标准（Standard for the Exchange for Product model data，STEP，ISO 10303）和 SGML 的整合。STEP 标准涉及来自各个行业的产品数据，并且为建模提供了扩展支持、自动存储方案生成、生命周期维护和其他管理工具。

在本章中，我们也涉及多种文本转换技术，我们称之为简单文本操作。我们首先讨论了用于预处理文档文本和为搜索和查询目的而产生一系列索引项的 5 种不同的文本操作。这 5 种文本操作是词汇分析、禁用词去除、词干提取、选择索引词和同义词典。前 4 个是与生成一个好的索引项集合直接相关的。第 5 个，同义词典的创建，与索引项关系分类层次的创建
251 更相关。然后这些关系可以通过能更好地适应用户需求的描述来扩展用户查询（手动的或自动的）。

最近，对于禁用词去除、词干提取和索引项选择给检索性能带来的潜在改进存在争论。实际上，也没有确凿的证据证明这些文本操作可以在检索性能上产生持续的改进。因此，现代检索系统可能不再使用这些文本操作。这个趋势的一个很好的例子就是某些 Web 搜索引擎对所有的词建立索引，而不管这些词的语法特性或者在文本中的角色。

而且，利用基于同义词典技术的自动查询扩展能否提高检索性能也是不清楚的。使用同义词典直接帮助用户进行查询描述同样也说不清楚。实际上，过去 Yahoo! 使用了一个索引项的分类层次结构向用户展示了项间的关系，这个现象也预示着基于同义词典的技术可能对于为现代数字图书馆系统开发的高度交互界面相当有用。

与之前不同的一个文本操作是压缩。之前的多种文本操作旨在以某种方式提高答案集的质量，而压缩文本的操作是为了减少空间、I/O、传输开销以及搜索时间。

为了能在信息检索环境中高效处理，压缩方法应该满足下面的要求：更好的压缩率、快速编码、快速解码、不用从头解码的快速随机访问，以及（更好地是）不需要解压缩文本而直接搜索。好的压缩率节约了辅助存储空间并减少了传输的开销。快速编码减少了由于将压缩引入系统所带来的处理开销。通常快速解压缩比快速编码更重要，因为在文档系统中文档只需要被压缩一次，而需要从磁盘上解压缩很多次。快速随机访问和直接搜索可以更高效地处理信息系统用户提交的多个查询。我们利用这些需求作为参数比较了多种压缩方案。

我们的讨论说明，基于词的半静态方法是将压缩引入现代信息检索系统的最佳选择。特别是，位霍夫曼、密集编码和 Re-Pair 是最有前途的变种。这些方法允许文本的随机访问、不用解压缩的高效直接搜索以及在 Re-Pair 情况下非常有价值的浏览文档集的方法。这些方法都获得了有吸引力的压缩比和解压缩时间。除了 Re-Pair 外，它们也提供了很好的压缩时间。另外，它们是基于词的，从而简化了把它们整合到 IR 系统的过程。

我们也讨论了最近一些利用文本结构优点的方法，因为这正在成为处理信息检索系统中文本的标准操作。

当前文本压缩的一个挑战是将索引压缩整合到 IR 系统中（参见第 9 章），因此，不只是文本而且索引也可以被压缩。这也引起了我们对同时实现旨在归纳文本集信息内容的压缩和索引的兴趣。在本章中，我们已经看到像 Re-Pair 的方法如何使用在文本集上构建的字典作 252
为浏览重要短语的工具。我们也看到了文本集的词汇表如何被用做索引也被用做基于词的压缩。我们也看到了，作为文本 0 阶模型一部分的词汇表，如何用来将复杂搜索分解为不同的文本单词，从而使得压缩文本可以比原始文本查找加快很多倍。其他一些压缩和检索的关系，特别是关于 BWT 的内容，会在第 9 章中讨论。

将压缩包括到信息检索系统中的一个难题是，随着文本集的演变如何维护压缩。用于创建半静态模型的原始统计，在文本集进行了比较大的改变后（例如，考虑新闻文档集）可能就不再适用了。定期重新压缩当然是一个选择，但是如何避免信息检索系统复杂的定期重新处理非常引人关注。另一方面需要注意的是，有些文档集随时间相对比较均匀，并且对文档集的第一个采样（例如 10M）计算的统计信息也能很好地压缩它之后的部分。

6.10 文献讨论

在 6.1 节中用到的文档模型基于 [905]。Web 元数据的详细介绍在 [982, 983] 中给出。关于标记语言的大部分信息和相关问题来自于万维网联盟（World Wide Web Consortium）（查看 www.w3.org）。关于 SGML 和 XML 更多的信息由 Goldfarb[633, 634] 给出。SGML 的其他引用是 [757, 1572]（特别是 SGML 的例子是从 [474] 中获得的）。有关于 HTML 的书有数百本。关于 HTML 4.0 的两个资源是 [474, 1656]。关于 CSS 的书是 [1027]。关于 XML、XSL 和 XLink 的信息查看 [1655, 1657, 1658]。关于 XML 及其相关语言的缺点和优点的讨论查看 [241, 413, 904, 930]。更多关于多媒体格式的信息可以在 [1005] 中找到。对于图像的情况参见 [1703]。

我们对词汇分析和禁用词去除的讨论基于 Fox[577] 的工作。对于词干提取的讨论基于 Frakes[581] 的工作。Porter 的词干提取算法在 [1292] 中提出。同义词典的讨论基于 Foskett[574] 的工作。然而这里我们并没有涉及同义词典的自动生成。这些讨论可以在第 5 章和 [1501, 1526] 中找到。另外一些在同义词典用途上的讨论在 [856, 1501] 中呈现。

关于文本压缩，有几本书可以参考。对于压缩在信息检索系统中最出色的引用是由 Witten、Moffat 和 Bell [1709] 写的。他们涉及了很多我们在这里提到的文本压缩技术，同时也介绍了图像和文本集索引的压缩。他们也给出了一些文本压缩方法的实现，例如霍夫曼和算术编码，这是称为 MG 的、用 ANSI C 编写的完整的压缩检索系统（http://www.cs.mu.oz.au/mg）。Bell、Cleary 和 Witten[174] 主要涉及统计方法，将重点放在了建模和编码任务上，包括半静态和自适应模型，以及自然语言建模。他们也关注字典方法。Moffat[1148] 最关注各种 253
编码技术和它们的有效实现。

Shannon[1452] 的基础论文被看做是信息论的诞生。霍夫曼编码最初由 [795] 提出。基于词的压缩在 [175, 184, 775, 1147] 中讨论。基于词的霍夫曼编码器可以在更庞大的 MG 系统找到。[769, 1445] 讨论了规范码。字节霍夫曼编码及其直接搜索能力，包括无空格单词模型，都在 [1158] 中讨论。换行 PPM 建模由 [173] 提出。还有多个压缩软件基于将算术编码附加到 PPM 模型上。相当高效的一个是 Shkarin 的 ppmdi[1469]，这个软件可以在 XMLPPM 的实现中下载，也可以在 PizzaChili 的网站上下载到（参考后面内容）。

第一篇关于算术编码的文章是 [1357]。其他参考文献是 [174, 1710]。一个开源的算术编码器可以在 http://www.cs.mu.oz.au/~alistair/arith coder 上下载。

密集编码和它们的快速搜索能力在［265，550］中详细地进行了研究。它们的名字来源于它们与标签霍夫曼编码（tagged Huffman codes）的关系，而后者现在已经很少使用了。［266］描述了密集编码的一个自适应版本，这个版本可以支持搜索。

Ziv-Lempel 压缩算法在［1795，1796］中提到。很多压缩软件都基于此，例如免费的 gzip（http://www.gzip.org）和 Zip(http://www.info-zip.org）软件。直接搜索 Ziv-Lempel 文本的实践工作以及相关软件 LZgrep 可以在 http://www.dcc.uchile.cl/～gnavarro/software 上找到。Re-Pair 由［981］首次提出。它的基于词的版本和短语浏览的用法在［1150，1663］中提到。

Burrows-Wheeler 变换在［302］中提到。之后它获得了大量的关注，例如［1192］和第9章中提到的压缩索引的作用。压缩软件 bzip2(http://www.bzip.org）是基于这个变换的。

存在多种结构化文本压缩方法。我们已经介绍了 XMLPPM[367]（免费软件在 http://xmlppm.sourceforge.net 中获得）、SCM[12] 和 LZCS[13]（这两个软件可以在 http://www.infor.uva.es/～jadiego/download 上下载）。更多的参考资料可以在这些论文中找到。

最后，http://www.data-compression.com、http://www.data-compression.info 和 http://datacompression.info 提供了一些与压缩相关的在线资源。用于压缩算法的标准文本语料是 Canterbury 语料库（http://corpus.canterbury.ac.nz）和 PizzaChili 网站（网站镜像包括 http://pizzachili.dcc.uchile.cl 和 http://pizzachili.di.unipi.it）。TREC 文档集也经常用于
254 作为压缩的评测文本（http://trec.nist.gov）。

查询：语言及属性

——与 Gonzab Navarro 合著

这一章涵盖了查询的主要方面，包括表达查询的不同语言、查询分布，以及 Web 的查询分析方法。

7.1 查询语言

提交给文本检索系统的查询通常有着不同的类型。用户可能生成的查询类型在很大程度上依赖于第 3 章所讨论的底层检索模型。所以，对于相同类型的查询，全文检索系统的回答可能与其他检索系统不同，包括基于关键词排序的检索系统（如 Web 搜索引擎），或者基于超文本模型的检索系统。第 9 章将解释系统是怎么处理用户查询的，本章将解释用户是怎么生成查询的。

正如之前章节所介绍的，我们需要区分*信息检索*和*数据检索*这两个概念，因为我们需要使用这两个概念区分不同类型的查询语言。如果是允许答案排序的查询语言，我们称之为信息检索语言；如果是不考虑答案排序的语言，我们称之为数据检索语言。正如第 3 章所讨论的那样，对于基本的信息检索模型，基于关键词的检索是主要的查询任务。而对定位于非信息检索的查询语言，排序的概念并不容易定义，因此我们将这类查询看做数据检索语言。而且，有些查询语言并不是给最终用户使用的，而是给更高层次的软件包用来查询在线数据库或者光盘文档的语言。在这种情况下，我们称之为*协议*，而不是查询语言。根据用户体验，通常可以使用不同的查询语言。比如，如果用户明确地知道感兴趣的信息是什么样的，那么检索任务将变得更加简单，甚至不需要答案排序。 255

一个重要的问题是，大部分的系统都试图通过内容（如语义）和文本的结构（如文本的语法结构）来找到相关文档。但是，由于任务本身比较困难，因此系统可能会找不到相关文档（参见第 4 章）。由于这个原因，出现了一些旨在提高查询实用性的技术。比如将查询用它的同义词集扩展，或者使用词典扩展，又或者采用词干提取的技术将同一个词的所有变化形式放在一起扩展，还有一些其他的技术。此外，那些频率很高且无意义的词（如英语里的 **“the”**），即*禁用词*，可以删除。这个主题在第 6 章中涉及。在这里，我们假设对所有查询已经完成了预处理。尽管这些操作在信息检索里是常见的，但许多操作对数据检索同样很有帮助。在需要的时候，我们将能够匹配查询项的词称为“关键词”，以区分那些不能与查询项匹配的词（因为这些词没有包含在索引里面）。

与查询类型相关的一个问题是，在信息系统中采用的*检索单元*的主体是什么。检索单元是指那些能够被检索到并作为查询答案的基本元素（通常是被检索到的一个基本元素集，有的时候也需要以相关性或者其他标准排序）。检索单元可以是文件、文档、网页、段落，或者包含查询答案的其他结构性单元。从这一点看，我们把检索单元简称为“文档”，这个简称也可以用做其他的含义（参见第 3 章）。

本章的组织方式如下。我们首先介绍使用基于关键词的查询语言表示的查询。这类查询是针对信息检索任务的，除了包括某些简单的单词和短语外，还包括可以操作文档集的布尔操作符。其次，我们将涉及包括更加复杂的查询的模式匹配。模式匹配的目的通

常是用更强的数据检索能力来补充关键词搜索。再次，我们将涉及针对文本结构的查询，它更加依赖于特定的文本模型。最后，我们将介绍用于互联网和光盘出版物的一些标准协议。

7.1.1 基于关键词的查询

查询是用户信息需求的形式化表示。最简单的查询是由关键词构成的，要搜索的是包含查询中关键词的文档。基于关键词的查询很受欢迎，因为这样的查询很直观、易于表达，同时也支持快速排序。因此，查询（在许多情况下）可以简单到只有一个词，虽然也可以包括由复杂操作组合成的多个词。在本章中，我们将单词查询（单个词或者多个词）称为基本查询。在7.1.2节所讨论的模式也视做基本查询。

1. 单词查询

256 在文本检索系统中，最基本的查询只有一个单词。假设文本文档本质上是由单词组成的长序列。尽管有些模型提出了更加一般的假设，但几乎所有的模型都允许以这种观点看待文本。有些模型也将单词分解成字母，这样就允许模式搜索，而非单词查询，这部分我们将在7.1.2节中讨论。然后，这些经过扩展的查询将检索到的单词集交给单词处理模块处理，完成同义词典扩展或者排序。

通常用一种十分简单的方式定义单词。假如字母表中包含的是字母和间隔符两类符号，那么单词就是一个被间隔符围绕的字母序列。更复杂的模型允许一些特定的字符，它们不是字母，也不能用作分隔符，比如单词“**on-line**”中的连字符。让文本数据库的管理员选择哪些字符是字母，哪些是分隔符，是一个很好的做法。

并不是漫无目的地将文本分解为单词，因为在自然语言里，单词包含了很多的语义。因此，许多模型（如向量模型）在单词概念上建立复杂的结构，并且单词查询是允许的唯一一种查询（而且，有些系统只允许从文档中抽取出一小部分单词）。

如果单词查询是析取形式的，那么查询产生的结果就是至少包含查询中一个单词的文档集。例如，在向量空间模型中（参见3.2.6节），结果集中的文档根据它们与查询的相似程度排序。为了支持排序功能，经常使用两种基于文档内单词出现频率的统计量。第一种称为“项频”，它统计单词在一篇文档内出现的次数。第二种称为“反比文档频率”，它统计出现某个单词的文档数量。第3章中详细讨论了单词的权重赋值策略。

另外一种查询的表达方式是合取形式，这种方式在Web搜索引擎中很流行。在这种情况下，只有当文档中包含了查询中的所有单词，这个文档才能匹配查询。当查询中的单个词能匹配的文档数量非常大时，这种形式就很有用。但有时，合取形式可能要求过于严格，可能导致很少甚至没有答案文档。在这种情况下，可以通过丢掉某些词来放宽限制要求。这样我们可能感兴趣的是保留能够一起出现的单词的子集（参见下面的邻近性）或者可能检出大量答案文档的单词子集。

另外，也可能要求系统提供某个单词在文本中出现的准确位置。在显示结果时，这些信息有助于在返回的文档片段中加亮显示单词的出现位置。

2. 上下文查询

除了单词查询能力外，许多系统也有能力在一个特定的上下文片段中搜索单词，即在其他单词附近搜索单词。也就是说，我们把文本中接近的物理位置作为上下文使用。查询中的单词彼此接近，可能意味着比那些彼此远离的单词有着更高的相似性。举例来说，我们可能想要构成短语，或者发现在文本中最接近的单词。

- **短语**：单个词查询的序列。一个短语等同于多个单词的序列。例如，假定查询 **"enhance retrieval"**，我们可以先搜索单词 **"enhance"**，然后搜索单词 **"retrieval"**。 257
  处理短语查询时，一般认为文本中出现的间隔符不需要与查询中出现的间隔符相同（例如，连续两个单词之间可以有两个空格或者只有一个空格），甚至可以完全不考虑不感兴趣的词。例如，在之前的查询例子中，可以匹配到这样一个文本 **"... enhance the retrieval ..."**。尽管在大多数的情况下，这个特性是很有帮助的，但并不是所有的检索系统都实现了这个特性。
- **邻近性**：短语查询的一个更加宽泛的形式是邻近查询。在这种情况下，给定一个单词或者短语形成的序列，同时给定它们之间允许的最大距离。例如，上面的例子可以表述成这样，两个查询单词应该在间隔四个词的范围内出现，因此下面这样的文本被匹配，**"... enhance the power of retrieval ..."**。这里的距离可以用字符或者单词来衡量，主要取决于系统。可以要求或不要求单词和短语以查询中出现的次序出现。

短语可以在一定程度上类似于以单个词的方式排序（详见第 3 章）。如果排序函数使用的参数不依赖于物理上的邻近位置，那么邻近查询也能以相同的方式排序。尽管并不知道怎样做才能更好地排序，但是物理位置的邻近性还是有语义上的价值。这是因为，在大部分的情况下，邻近性就意味着查询单词出现在相同的段落中，因此在某些方面是相关的。

3. 布尔查询

将关键词查询组合起来最古老的（至今仍然大量使用的）方法是使用布尔操作符。布尔查询是一种由原子（即基本查询）和布尔运算符组成的语法。原子用来检索文档，布尔运算符作用于操作对象（检出的文档集）并指明合适的结果文档集。因为这种查询策略一般是可以组合的（如运算符可以与其他运算符的结果组合），可以很自然地定义一棵查询语法树，语法树上的叶子结点对应于基本查询，而内部结点对应于布尔运算符。查询语法树在各个文档集上进行代数运算（查询的最终答案也是一个文档集）。这很像算术表达式的语法树，在算术表达式的语法树中，数字和变量是叶子结点，而算术操作是内部结点。图 7-1 展示了一个例子。

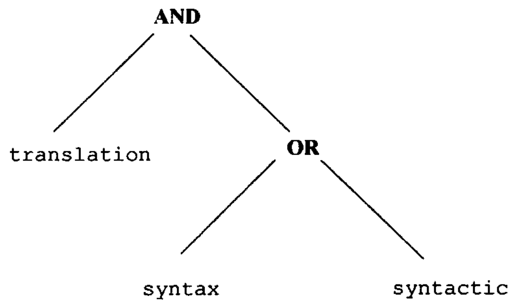


图 7-1　查询语法树的一个例子。这个查询检索所有包含单词 **"translation"**，同时包含单词 **"syntax"** 或者单词 **"syntactic"** 的文档

给定两个基本查询或者布尔子表达式 e_1 和 e_2，通常使用的运算符有：

- **OR**：查询（e_1 OR e_2）选择所有满足 e_1 或 e_2 的文档。重复的文档会被删除。
- **AND**：查询（e_1 AND e_2）选择所有同时满足 e_1 和 e_2 的文档。
- **BUT**：查询（e_1 BUT e_2）选择所有满足 e_1 但不满足 e_2 的文档。需要注意的是，经典的布尔逻辑使用的"NOT"运算符。当文档不满足 e_2 时，（NOT e_2）是合法的。在这种情况下，所有不满足 e_2 的文档都应该被检出，这可能检出大量的文本，并且很可能不是用户想要的结果。相反，BUT 运算符将可检出的文档全集限制为满足 e_1 的文档[⊖]。 258

除了选取适当的文档外，系统也可以完成：1）将文档以某一标准排序；2）加亮显示文

⊖ 注意到关系运算符也有同样的问题，和关系代数一样需要避免"不安全的"表达式。不安全的表达式直接或者间接引用元素的全集，例如 NOT 算子所做的。

档中出现的查询词；3）将答案集作为基础重构查询，并且进行反馈。这些信息也将作为结果集的一部分提供给用户。

典型的布尔系统不对检出的文档进行排序。一个文档要么满足布尔查询（被检出），要么不满足布尔查询（不会被检出）。这是很大的限制，因为这种系统不允许用户查询和文档之间存在部分匹配。为了解除这种限制，检索条件必须放宽。例如，部分满足 AND 条件的文档也可以被检出。

众所周知，没有接受过数学训练的用户发现布尔运算符的意义很难掌握。考虑到这个问题，研究人员提出了一种“模糊布尔”运算符集。想法是可以放宽 AND 和 OR 运算符的意义。模糊运算符不要求满足所有条件（AND）或者满足至少其中一个条件（OR），而是返回满足部分条件的结果。模糊运算符检出满足某些条件的文档。在这种情况下，AND 运算符要求满足比 OR 运算符更多的条件。而且，如果满足较多的查询条件，文档将被排在更前的位置（参见第 3 章）。

7.1.2 非关键词查询

259 本节将讨论表示能力更强的查询语言，即模式匹配和自然语言。

1. 模式匹配

在这一部分中，我们讨论更强大的查询描述（基于模式的概念），这种查询描述允许检索满足某些属性的文本片段。这些数据检索查询对语言学、文本统计和数据抽取是很有用的。这种查询返回的结果可以用之前讨论过的组合机制生成短语和邻近查询，构成所谓的基本查询。基本查询可以用布尔表达式组合起来。在这种意义下，可以将这种数据检索能力看做是信息检索的增强工具。然而，将更难对模式匹配表式的结果文档进行排序。

模式是一组在文本段中发现的语法特征。那些满足某个特定模式的文本段称为“匹配”这个模式。我们对那些包含能够匹配给定搜索模式的文本段的文档感兴趣。每个系统都允许指定一些模式类型，可以是非常简单的类型（如一些词），也可以是相当复杂的类型（如正则表达式）。模式集允许使用的类型越强大，用户可以描述的查询越多，而搜索函数的实现也越复杂。最常用的模式类型有以下几种。

- **词（words）**：必须是文本中单词的字符串（字符序列）（参见 7.1.1 节）。这是最基本的模式。
- **前缀（prefixes）**：必须是文本中（一个或多个）单词开始部分的字符串。例如，给定前缀 “**comput**”，所有包含有如 “**computer**”、“**computation**” 等单词的文档都会被检出。
- **后缀（suffixes）**：必须是文本中（一个或多个）单词结尾部分的字符串。例如，给定后缀 “**ters**”，所有包含 “**computers**”、“**testers**”、“**painters**” 等单词的文档将被检出。
- **子串（substrings）**：出现在文本中的单词的子串。例如，给定的子串是 “**tal**”，所有包含 “**coastal**”、“**talk**”、“**metallic**” 等单词的文档将被检索到。这种查询可以限制在单词内部查找子串，或者更进一步地在文本的任何地方查找子串（在这种情况下，查询不局限于查找字母序列，而可以包含单词分隔符）。例如，“**any flow**” 将匹配短语 “**... many flowers ...**”。
- **范围（ranges）**：能够匹配以词典序位于一对字符串之间的字符串。词典通常是有序的，这就引出了称为*词典顺序*的字符串顺序（事实上是单词在词典中排列的顺序）。

例如，在单词“**held**”和“**hold**”之间的范围将检索到诸如“**hoax**”和“**hissing**”这样的字符串。

- **容错（allowing error）**：在一个错误阈值范围内的单词。这种搜索模式检出的是文本中所有与给定单词“相似”的单词。相似的概念可以有多种定义。一般假设模式或者文本可能包含错误（来自打字、拼写或光学字符识别软件等），那么查询不仅应该检出给定的词，而且应该检出给定词的各种错误形式。尽管已经有许多模型用来定 260
义单词之间的相似度，但在文本检索领域广泛接受的是 Levenshtein 距离，或者称为编辑距离（edit distance）（参见 6.5.3 节）。这个距离在模型误差上要优于其他更加复杂的方法，如 Soundex 系统。因此，容错查询指定匹配一个单词允许的最大错误数（如允许的最大编辑距离）。这个模型也可以扩展到搜索子串（不只是单词），检索在搜索模式允许的编辑距离内的任意文本段。在这个模型中，如把“**flower**”分开成“**flo wer**”的打字错误仍然可以在一个错误的距离内被检测到。需要注意的是，在只考虑单个词这种更加严格的情况下，将检测不到这个打字错误（因为不论“**flo**”还是“**wer**”，与“**flower**”的编辑距离都不是 1）。在计算生物学中，使用这种距离模型的变体来搜索 DNA 和蛋白质序列。
- **正则表达式（regular expressions）**：某些文本检索系统允许用正则表达式搜索。正则表达式是一种更加普通的模式，是用简单的字符串和以下操作构建的。
  - □ 联合（union）：如果 e_1 和 e_2 都是正则表达式，那么（$e_1 | e_2$）与（e_1 or e_2）匹配相同的元素。
  - □ 连接（concatenation）：如果 e_1 和 e_2 都是正则表达式，（$e_1\ e_2$）表示出现 e_1 后面立即跟着出现 e_2（因此，简单的字符串可以被认为是字符串中各个字母的连接）。
  - □ 重复（repetition）：如果 e 是一个正则表达式，那么（e^*）匹配 e 出现 0 次或者多次连续出现的序列。

  举例来说，考察一个查询“**pro(blem | tein)(s | ∈)(0 | 1 | 2)***”（这里∈代表空字符串）。这个查询可以匹配单词“**problem02**”和“**proteins**”。正如前面的情况，这种匹配可以限制到包括一个完整的单词，或出现在单词内部，或匹配任意文本段。正则表达式也可以与前面几种类型的模式组合，允许容错地搜索正则表达式。
- **扩展模式（extended patterns）**：使用对用户更加友好的查询语言来代表正则表达式的某些常见形式。扩展模式是正则表达式的子集，由更加简单的语法来表示。检索系统在系统内部将扩展模式转化为正则表达式，或者使用特定的算法搜索。每个系统支持自己的一套扩展模式，因此不存在形式化的定义。下面是一些在许多新系统中发现的扩展模式的例子。
  - □ 字符类型，即与一组字符相匹配的某些模式位置。这种特征包括：区分大小写的匹配、使用字符范围（如指定某些字符必须是数字）、补足（如某些字符必须不是字母）、枚举（如一个字符必须是元音）和通配符（如某些位置上可以匹配任何字符）等。
  - □ 条件表达式，即模式的一部分可以出现或者不可以出现。 261
  - □ 通配符可以匹配文本中的任意序列，如任何以“**flo**”开始并以“**ers**”结尾的单词，既可以匹配“**flowers**”也可以匹配“**flounders**”。
  - □ 组合可以允许模式有些部分精确地匹配而另外一些部分容错地匹配。

2. 自然语言

将模糊布尔模型更推进一步，AND 和 OR 之间的区别可以完全模糊。在这种情况下，

查询变成将用户感兴趣的单词和上下文查询进行简单枚举，与查询中任何部分匹配的文档都将被检出，同时那些与查询中越多部分匹配的文档将给予更高的权重。通过让用户描述不希望检索到某些词来得到否定形式，因此，那些包含这些词的文档将在排序计算中排到后面。可以选定一个阈值，这样权重非常低的文档将不会被检出。在这种策略下，我们完全消除了布尔运算符的影响，进入到了自然语言查询的领域。事实上，可以将布尔查询看做是自然语言查询的简化抽象。

这个模型的使用过程中产生了许多新问题，特别是，如何对查询对应的文档正确地排序。搜索条件可以使用另一个模型进行重新描述。其中，文档和查询都看做“带权重”的向量（每一个坐标对应一个感兴趣的关键词或者可以是文本中出现的一个单词），而查询也用完全相同的方法看待（上下文查询不在这种情况内）。因此，查询在模型内部被转化为一个带权重的向量，检索的目标就是要检出所有那些与查询接近（接近程度需要在模型中进行定义）的向量（文档）。若想要彻底地讨论这部分内容，可以参见第3章关于索引项权重和向量模型的部分。这就产生许多有趣的潜在应用。比如一篇完整的文档也可以作为一个查询（因为文档也是一个向量），这可以很自然地使用相关反馈技术（即用户可以从查询返回的结果文档中选择一篇文档，并将它作为新的查询提交给系统，从而检出与被选择的文档相似的文档集）。这个模型的算法和那些基于模式搜索的算法完全不同（甚至有可能不需要搜索文本中的每个单词，而是希望从每篇文档中抽取出一个较小的、具有代表性的关键词集合）。

7.1.3 结构化查询

到这里，我们已经可以将文本集看做是一个可以在上下文内容中查询的文档集。然而这个模型不能利用某些已经普遍采用的新文本特征，比如文本的结构。文本集往往有一些内在的结构特征。基于结构（不仅是内容）的文本查询正在成为一种非常有吸引力的趋势。HTML等用于描述文本结构的标准语言进一步推动了这种趋势。

在查询中同时使用内容和结构可以生成更强大的查询，这种查询比起单独使用内容的查询和单独使用结构的查询更具有表达力。通过使用一种整合了这两种查询的查询语言，可以改进文本数据库的检索质量。

262

这种机制建立在基本查询之上。首先选择一组在内容上满足特定约束的文档集（文档必须包含指定的单词、短语或者匹配指定的模式）。在此之上，可以对文档中结构性元素（如章、节等）使用包含、邻近或其他结构约束。可以在结构化查询上建立布尔查询，以便组合那些结构化查询检出的文档集。结构化查询构成了布尔语法树的叶子结点（参见图7-1中的例子）。另一方面，结构化查询本身也有一套复杂的语法。

在本节中，我们将分开讲解文本数据库中发现的各种结构类型。图7-2描绘了几种不同的结构类型。尽管结构化查询语言应该也可以支持排序，但这还是个未解决的问题。

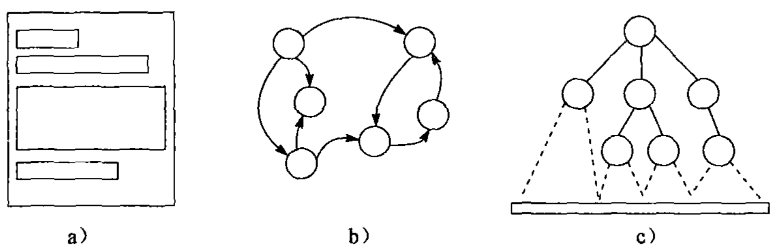


图7-2 三种主要的结构：a）表单型的固定结构，b）超文本结构，c）层次结构

接下来很重要的就是要区分文本可以拥有的结构信息以及可以被查询出来的结构信息。一般而言，自然语言的文本可以含有任何想要的结构信息。然而，不同的模型只允许查询这些真实结构中的某一部分。如果我们说可以查询的结构是有限制的，那么这意味着文本中遵循这些结构限制的部分可以查询出来，即使文本中可能含有更多的结构信息。例如，在一篇文章中，节与子节之间可能构成嵌套关系，但是查询模型不能接受递归结构。在这种情况下，我们不能查询嵌套在其他节中的子节，即便这些内容就存在于文本中。

1. 固定结构

文本中允许的结构传统上是很受限的。文档中含有一个固定的字段集，就像一个填满内容的表单。每个字段内有一些文本。有些字段并非出现在所有文档中，它们也很少以任意顺序出现，或者在整篇文档中重复出现，文档中也很少含有不在任何字段内的文本。这些字段不能嵌套或者重叠。在这些字段上允许的检索活动需要严格限制，只有在一个给定的字段上才能找到给定的基本模式。现在大部分的商用系统都使用这个模型。 263

当文档集有固定的结构时，这个模型是合理的。例如，归档的邮件可以看做一个邮件文本的集合，每个邮件都有发送者、接收者、日期、主题以及正文字段。这样，用户就可以搜索发送给特定某个人、主题为"**football**"的邮件。然而，这个模型并不能充分描述 HTML 文档中的层次结构信息。

如果足够精确地将文本分到各个字段，那么某些字段的内容甚至可以解释为文本之外的内容，如数字、日期等，因此可以在这些字段上生成不同的查询（如日期字段上的月份范围）。不难看出这个想法可以很自然地与关系模型联系起来，每个字段对应于数据库表格中的列。如果将数据库看做文本，那么将可以使用比在关系数据库系统中更强大的能力查询文本型的字段。另一方面，关系数据库可以更好地使用与构建数据库时指定的数据类型相关的知识，建立更有效的索引。在过去的几年中，已经提出了一些混合两种模型的方法，主要的问题在于不能达到最优的性能，因为文本通常是和其他类型的数据一起存储的。虽然如此，已经有多种方法将结构化查询语言（Structured Query Language，SQL）扩展到允许全文检索。在这些方法中，我们将在 13.6.2 节中介绍 XML 全文查询语言，在 7.1.4 节中介绍 SFQL 查询语言。

2. 超文本

超文本可能代表与结构化能力相反的趋势。超文本是一张有向图，图中的结点上可能代表某些文本，链接代表两个结点之间的关系，或者代表结点内部的位置关系。随着 Web 的爆炸式增长，超文本也受到了大量的关注。事实上，Web 成为遍布全球的、庞大的超文本类型的数据库。

然而，从超文本中检索信息开始演变成了一个纯粹的导航型行为。也就是说，用户不得不人工地跟随链接遍历超文本的结点，以便搜索到他们想要的信息。他们也不可能根据超文本的结构查询信息。即便是在 Web 上，人们只能搜索结点上的文本内容，而不能根据链接结构搜索信息。

一项有趣的研究将在 Web 上的浏览行为和搜索行为结合起来，这项研究是 WebGlimpse [1078]。它将经典的导航行为与搜索当前结点附近内容的能力结合起来。如今，已经出现了一些查询工具，能够完成基于超文本内容以及其结构查询信息的目标。

3. 层次结构

在固定结构和超文本结构之间存在着一个中间的结构化模型，这就是层次结构。它是对文本的递归式分解，并且对于许多文本数据集来说，这也是一个很自然的模型（如书籍、文

第一项研究集中在基本统计上，如查询出现频率、项频，以及每个查询包含的单词数量。自从 20 世纪 90 年代以来，这些统计并没有发生很大的变化。在 1997 年和 1998 年 [1479，826]，平均查询长度大约是 2.4。在使用 1997 年数据的一项研究中 [826]，单个词的查询占 31%，而在使用 1998 年数据的另一项研究中 [1479]，占 26%。Jansen 和 Spink [824] 声称单个词查询的百分比减少是一种趋势。相反地，Jansen 等人的另一项研究 [823] 发现，三个词查询的百分比从 1998 年的近 28%增加到 2002 年的 49%。最近，Jansen[819] 完成了一项研究，这项研究在 2005 年 5 月收集了来自 dogpile.com 检索引擎的 150 万个查询，发现查询的平均长度是 2.8 个词，最长的查询使用了 25 个词。然而，这些结果大部分都没有被很好地解释过，因为在大部分情况下，查询日志规模都比较小，并且在检索引擎中的抽样过程也并不完全清楚。Skobeltysyn 等人 [1487] 在研究查询缓存时给出了在一个较大的数据集上的统计信息。这个大数据集包含了来自 2007 年英国 Yahoo! 的 1.85 亿
267 个查询。表 7-1 列出了最近两篇参考文献中有关查询长度的分布。

表 7-1 查询长度的舍入百分比。数据来自 [819]，基于 dogpile.com 的查询日志，以及来自 [1487]，基于英国 Yahoo! 查询日志

长度	Dogpile(2005 年)	Yahoo!(2007 年)	长度	Dogpile(2005 年)	Yahoo!(2007 年)
1	18	25	5	6	4
2	32	35	6	3	1
3	25	23	>7	3	1
4	13	11			

查询可以看做是文本中的词，服从有偏置的分布。实际上，查询的频率满足参数为 α 的 Zipf 法则（参见 6.5.2 节），也就是说，频率最高的第 i 个查询的出现频率是 $O(i^{-\alpha})$（参见图 7-4）。参数 α 的取值范围从 0.6～1.8[428，123，104]，造成取值不同的原因可能是语言和文化的差异。无论如何，比起 Web 文本，这种差异要小得多。在 Web 文本中参数 α 的取值接近于 2。相同的图上显示了查询项的分布，比完整的查询分布偏差更大些（也就是说 α 更大），而与 Web 上索引项的分布更加相似。

网页中的词频与查询中的词频之间的标准相关性有差异，但并不高，变化范围为 0.15 [123] ～0.42[104]。这就是说，网页内容中的词也服从 Zipf 分布，尽管两者的分布看起来很相似（参见图 7-4），但阶有很大的不同（图 7-5 描述了这个事实）。这意味着用户搜索的信息和用户发表在网页上的信息存在着很大的差异。

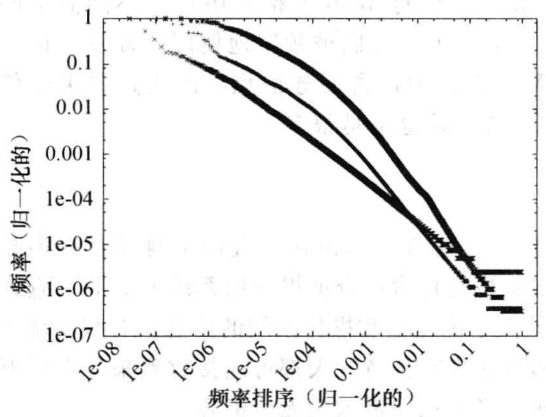


图 7-4 英国 2006 年 Web 样本中查询（底部曲线）、查询项（中间曲线）和索引项的归
268 一化频率 [104]

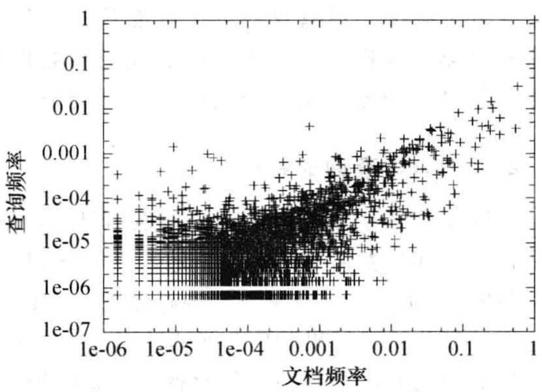


图 7-5 对于词汇表中的每个词，图中横轴为相对文档频率，纵轴为相对查询频率 [104]

- **通用命令语言（Common Command Language，CCL）**：CCL 是 NISO 根据 Z39.50 协议的提议的协议（Z39.58 或 ISO 8777）。该提议定义了 19 种可以交互使用的命令。尽管很少有产品使用这种协议，但是在欧洲，该提议还是很流行的。该协议基于经典的布尔模型。
- **光盘只读数据交换（Compact Disk Read only Data exchange，CD-RDx）**：CD-RDx 使用客户机/服务器架构，并在大部分现有的平台上实现。客户端是通用的，而服务器是由光盘发行商设计并提供的，作为光盘的一部分。该协议允许使用固定长度的字段、图像和音频，并已经被某些美国国家机构支持，如 CIA、NASA 和 GSA。
- **结构化全文查询语言（Structured Full-text Query Language，SFQL）**：SFQL 基于 SQL 语言，也是客户机/服务器架构的协议。SFQL 已经被航空航天领域（如美国空运协会/飞机工业协会）接纳为标准。文档对应于关系表中的行，并可以使用 GSML 语言进行标注。该语言定义了答案的格式，有一个头部和一个变长的消息区，但没有定义任何特定的格式或者标记。例如，一个 SFQL 查询：

  Select abstract from journal. papers where title contains "text search"

  该语言支持布尔和逻辑操作、同义词典、邻近操作和某些特殊的字符如通配符和重复。

  例如：

  ... where paper contains "retrieval" or like "info %" and date>1/1/98

  与 CCL 或 CD-RDx 协议比较，SFQL 协议是基于关系模型的，虽然这种模型对于文档数据库并不总是最佳选择，但更加通用和灵活。

使用这些语言的一些系统将在第 16 章和第 17 章中介绍。 266

7.2 查询属性

人们为了查找信息，通常需要执行各种检索任务。检索任务可以从查找特定信息到探索性地浏览信息 [1082，1547]，因为人们在不同时间和不同环境中的检索需求是不一样的。在这里，我们使用环境来指代用户的使用环境。因此，检索查询可以跨越不同的需求，从查找事实性的信息，如"巴西的首都是哪里?"[⊖]到收集关于一个主题的信息，如"如果在巴塞罗那度假，我们应该看些什么?"[⊜]再到浏览文档集，如浏览 Flickr 图像或者科技论文集。

然而，有的时候，在执行检索任务的同时，执行一些并不总是与信息需求相关的任务也是需要的。事实上，在近些年里，有些研究人员提出了针对 Web 上检索用户的不同需求的特定分类标准。尽管这些分类标准并不十分理想，但是由于它们被频繁地使用和提及，记住这些分类标准也是十分重要的。我们不会在非 Web 查询中包含任务信息，因为这些关于任务的信息是依赖于实际应用的，并且关于这种情况的文献也不是很多。

7.2.1 Web 查询的特征

Web 搜索引擎记录了用户检索信息时的查询信息。这些信息包含了查询本身以及许多相关属性，称为查询日志。典型地，一个查询日志可能也会包含该查询提交给系统的时间、查询来源的 IP 地址、用户浏览器内存储的 cookie，以及关于浏览器和操作系统的信息。有些系统也会记录哪些检索结果和广告被点击的信息。由于搜索引擎在任意一天都可能接收到来自上千万用户提交的数亿个查询，因此查询日志成为一种理解各种用户需求的无价资源。

⊖ 答案既不是里约热内卢也不是圣保罗。

⊜ 关键词可以是 Gaudi、Miro 和 Picasso。

章、法律文档，以及结构化的程序代码等)。图 7-3 显示了这种结构的一个例子。

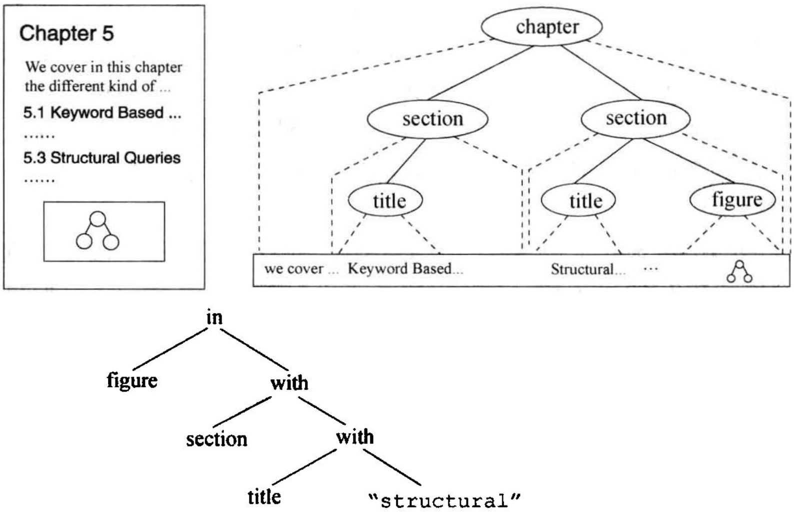


图 7-3 层次结构的一个例子：书的页、概要视图，以及检索图片的查询分析树

另一方面，从超文本结构简化到层次结构，就可以使用快速算法来处理查询。一般规律
264 是，模型的能力越强，模型实现起来的效率就越低。

在第 13 章中，我们将详细地讨论在层次结构中检索信息的问题，特别是基于 XML 文档中的层次结构，同时包括已提出的检索 XML 文档的所有查询语言（参见 13.6.2 节）。

7.1.4 查询协议

在本节中，我们简要介绍一些已经在软件应用程序中默认使用的文本数据库查询语言。其中一些已经作为查询光盘数据的标准，或者作为查询图书馆或者联机公共检索目录（OPAC）系统的中间语言。由于这些语言不是为人类使用而设计的，因此我们称之为协议，而不是语言。关于协议的更多信息可以在第 16 章和第 17 章中找到。其中一些内容是由于历史原因或者完整性的要求包含在其中。其中最主要的协议是：

- **Z39.50**：1995 年由 ANSI 和 NISO 组织批准为标准协议。该协议用来在客户和主机数据库管理员之间使用标准界面查询书目信息，而不管客户端用户界面的外观以及主机数据库的查询语言是什么。假设主机数据库是一些固定字段的文本数据集（这要比平时更加灵活）。也可以在其他地方使用这个协议，如 WAIS 系统在内部使用这个协议。该协议不仅规范化了查询语言及其语义，并且还规范了客户端和服务器
265 建立会话、通信和交换信息等的方法。尽管起初的设想只是为了操作书目信息（使用 MARC 格式），但该协议也扩展到查询其他类型的信息。
- **广域信息服务系统（Wide Area Information System，WAIS）**：WAIS 是万维网发展之前的 20 世纪 90 年代初期很流行的一组协议。WAIS 协议的目的是成为一个网络发布协议，并能够通过互联网查询数据库。

  在光盘发布领域，有许多种查询协议。这些协议的主要目标都是提供“光盘互换能力”。这意味着原始信息提供者和最终用户之间的数据通信更加灵活。还可以显著地节约成本，因为允许访问各种信息，而无须购买、安装以及为不同数据检索应用训练用户。我们简要地介绍以下三种协议。

一个重要的问题是：服从 Zipf 法则是因为 1）有两组用户，一组用户询问热门的查询；另一组用户询问独特的查询；还是 2）所有用户都可以询问两种查询？在近期的文章中，Goel 等人 [630] 表明幂律，特别是长尾现象，是因为后一种情况。

搜索引擎日志也记录了看过的结果网页的数量，以及查询后选择的网页数量。当很多用户通过增加或删减词来精化他们的查询时，实际上大部分用户只会查看不到两页结果（即户不会查看在返回结果中排名 20 位以后的网页）。表 7-2 列出了 4 个不同的搜索引擎在不同范围和不同语言上的比较数据 [823，1479，1519，1587，1713]。

表 7-2　4 个不同搜索引擎的查询统计

测度	AltaVista（1998 年）	Excite（2001 年）	AlltheWeb（2001 年）	TodoCL（2002 年）
平均查词数	2.4	2.6	2.3	1.1
平均用户查询数	2.0	2.3	2.9	—
查询平均答案页面数	1.3	1.7	2.2	1.2
布尔查询	<40%	10%	—	16%

除此之外，正如经验研究 [1479] 显示的，答案网页的平均点击数非常低，大约每个查询只有 2 次点击。在 Excite[1521，1519] 和 AlltheWeb[823] 上完成的进一步查询研究以及两者之间的比较研究 [1520] 都表明，在过去的几年里，用户查询的焦点从休闲转移到电子商务，详细的讨论将在 7.2.4 节中展开。在文献 [168] 中，作者分析了非常大量的网页查询日志。这些查询日志是由数以千万的用户通过 AOL 搜索 Web 时提交的查询构成的。在这项研究中，查询日志被分为几个查询组，分别在一天的不同时间内提交给系统。分析结果突出显示了在不同查询组中，主题性分类查询的受欢迎度和独特性的变化情况。

7.2.2　用户搜索行为

当搜索信息时，用户行为可以通过导航图进行可视化。图 7-6 举例说明了一张用户行为的导航图。在这张图中，在不同状态之间的转换值表明了用户选择那条路径的比例。这个值等价于从一个状态出发转移到另一个状态的概率。在图 7-6 中，我们只画出了转移概率大于等于 1%的边。另外，在每个状态内的数值代表了用户停留在那个状态的概率。从图 7-6 中我们可以推导出以下的结论：

- 较少使用高级搜索（但我们必须拥有）。
- 用户很少精化他们的查询。
- 用户很少浏览目录。 269

这意味着，用户采用了一个试错的信息搜索策略，而不是设法构造更好的查询。

7.2.3　查询意图

在 Web 出现以前，用户查询都是与搜索感兴趣的信息相关联的。搜索通常是在一个专门的环境中完成的，访问和使用搜索系统不一定是免费的。因此，搜索工具的设计也总是定位于帮助用户写出好的查询，这就意味着采用的搜索语言通常都比较复杂。Web 的出现彻底地改变了这种现象。用户使用搜索引擎，不仅是为了查找信息，而且也是为了达到其他目的。

第一个并且最受欢迎的 Web 查询分类是由 Broder 提出的 [268]。他将查询分为 3 类：信息型（informational）、导航型（navigational），以及事务型（transactional）。在导航型查

询中，用户的目的是查找想要浏览的网站。一个典型的例子就是用户准确地回想起哪些查询可以直接引导到他们感兴趣的网站——这种习惯可以代替书签的功能。在事务型查询中，用户的目的是执行交互任务，如下载软件、预订场所，或者购买商品。另外，需要注意的是查询意图可能是有歧义的。举例来说，假如用户需要查找他们喜欢的歌手的信息，那么他们是要查找歌手的个人简介，还是官方网站或者一首歌？Broder 通过在线调查的方式估计出了这 3 种查询类型的百分比（数千人参与了调查，10％的反馈率，结果偏向于回答调查的用户），同时人工分析了数百个不涉及性的查询。结果表明信息型查询约占 39％～48％，导航
270 型约占 20％～25％，事务型为 30％～36％。

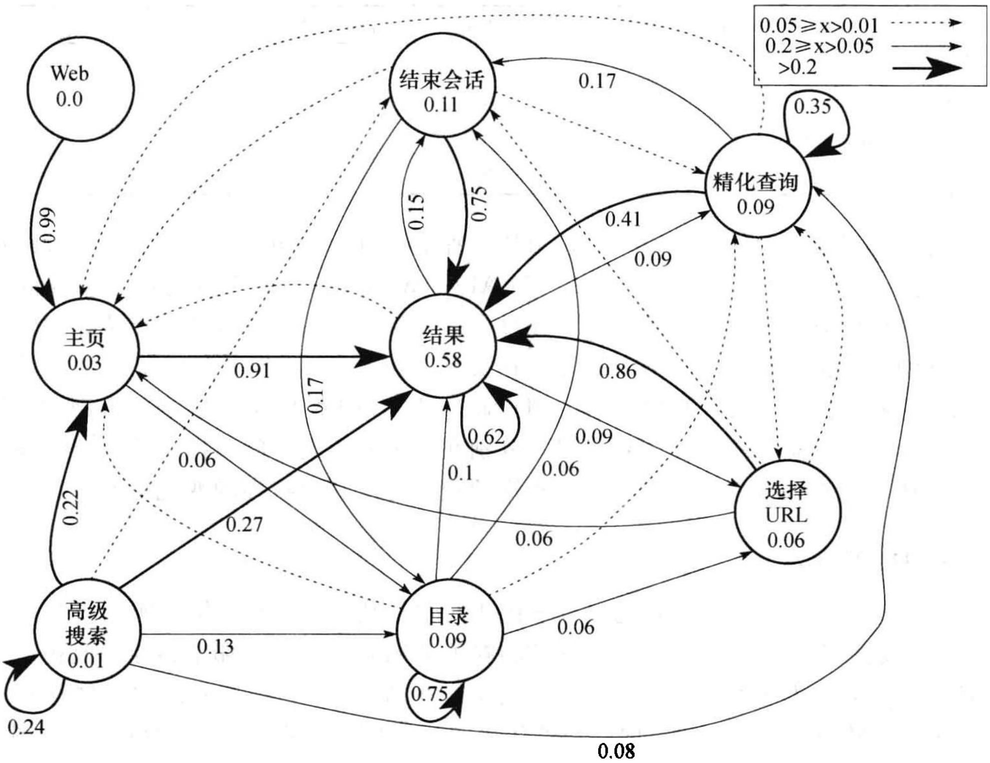


图 7-6 在搜索引擎中用户行为的状态图示例 [88]

之后，Rose 和 Levinson 完善了 Broder 的分类法 [1382]，他们为信息型查询和事务型查询定义了一些称为“在线资源”的子类。他们也注意到，有些查询可以被分到同一个大类下的多个子类中。如今，导航型查询的比例已经比先前提高了很多，据估计已经占到 40％。而对于信息型查询，他们发现，先前这类查询的比例超过 60％，但现在应该低了很多。

近期的研究主要集中在自动预测查询意图上 [93，822，872，997]。大部分的研究都使用机器学习方法，根据不同查询的属性来预测查询意图，如查询中锚文本的词分布、点击行为，以及查询长度。还可以使用相关属性，如被点击网页的文本和与结果相关的返回片段的文本。Baeza-Yates 等人 [93] 发现，信息型查询的预测准确率要比非信息型查询的预测准确率高很多。除此之外，正如所料，他们得出存在歧义的查询是很难预测用户意图的。主要的问题在于，由于搜索的上下文经常是未知的，因此许多查询自身就是有歧义的。

导航型查询的另一个问题就是所谓的*重新查询*（refinding queries）[852，853，1569]。

这类查询是指用户重复性地提交同一个查询，并且总是点击同一个返回结果。这个问题形成了这样的情况，用户用查询而不是书签来保存网页（例如一个会议的缩写）。Piwowarski 和 Zaragoza[1280] 表明，对于一部分查询会话（24%），可以用机器学习方法以很高的准确率（超过 90%）预测下一次的点击。Teevan 等人 [1568] 也获得了类似的结果（40%）。 271

正如第 5 章中讨论的那样，研究人员试图利用查询日志行为来改进排序算法，同时也尝试根据用户以往的行为预测用户的个人行为以及偏好。接下来的部分将探索后一种情况。

7.2.4 查询主题

查询也可以根据查询的主题分类，而不考虑用户查询意图。举例来说，在之前一个有歧义的查询例子中（搜索歌手），其主题是音乐。

过去，研究人员完成了很多查询主题分类研究。特别是，Spink 和 Jansen 完成了一系列关注于查询流不同特性的研究，例如主题是怎样随着时间或者地理位置变化的 [819，820，823，824，826，1519，1520，1521]。这些研究是通过人工分析查询日志来完成的。考虑到查询主题的变化，他们发现与娱乐和性相关的查询从 1997 年的高于 36%的比例下降到 2001 年的约 15%，而同时与商务相关的查询，从 13%增长到将近 25%。后来的研究 [819] 显示了这种变化更加显著，表 7-3 列出了这种变化。值得注意的事，因为查询日志是用不同方法抽样的，所以结果也只是（较好的）估计。

表 7-3　1997—2001 年在搜索引擎 Excite 上的主题变化，来自 from E-Sex to E-commerce：Web search changes，Computer 35(3)，pp：107-9(Spink，A.，Jansen，B. J.，Wolfram，D. Saracevic，T. 2002)，©2002 IEEE

排名	主题	1997 年	2001 年
1	Commerce，travel，employment，or economy	13.3	24.7
2	People，places，or things	6.7	19.7
3	Non-English or unknown	4.1	11.3
4	Computers or Internet	12.5	9.6
5	Sex or pornography	16.8	8.5
6	Health or sciences	9.5	7.5
7	Entertainment or recreation	19.9	6.6
8	Education or humanities	5.6	4.5
9	Society，culture，ethnicity，or religion	5.4	3.9
10	Government	3.4	2.0
11	Performing or fine arts	5.4	1.1

替代手动分类查询主题的方法是使用自动分类技术。Shen 等人 [1457] 收集了搜索引擎产生的最佳返回结果，并将结果映射到开放目录项目（Open Directory Project，ODP）对应的类别。结果很好，在 63 个类别上的 F 值达到了 0.45 左右。后来，Baeza-Yates 等人 [93] 使用各种查询属性将查询主题自动分类到 ODP 的类别中。结果显示分类的准确率依赖于不同的主题。这项研究的一个有趣的副产品是基于聚类技术的半自动分类方法，这个方法对于构造大规模训练数据集和评估数据集是很重要的。结合无监督技术和监督技术的结果可以在文献 [169] 中找到。 272

近来，Broder 等人 [276] 发表了一种以极高的准确率把短的/罕见的查询分到 6000 个商业类别中的方法。这是很有用的，举例来说，考虑到所有罕见查询量超过了搜索引擎查询

流量的一半以上（由于Zipf分布的长尾效应），为了改进与罕见查询相关的广告质量，这就是一个重要的问题，参见11.10.1节。他们根据每个类别的文档使用不同的文本分类器。对每个查询，他们通过一个给定的Web搜索引擎检出最靠前的前k篇返回文档，利用投票算法将该查询分到最优的一个或多个类别。如果考虑到在长尾中处于尾部的查询通常是最难处理的，那么返回结果是相当不错的。

正如在查询意图预测中的情况，查询不仅在意义上存在歧义，而且还可以被分到多个主题中，特别是当有不同类型的文档时（如一个查询不仅与政治相关，而且也与新闻相关）。少量的文章涉及歧义检测。例如，Song等人［1503］利用一个较大的样本集估计出大约有16%左右的查询是存有歧义的。

7.2.5 查询会话与任务

分析查询时面临的一个很重要的问题就是判定用户的查询会话。早期的工作利用固定的时间片定义会话。举例来说，一个查询会话可能是由同一个用户在30分钟的时间间隔内提交的所有查询组成的。然而，这样的定义存在两个问题：1）会话可能更长；2）在这个会话中用户可能有多个目的。因此，最好能够区分基于时间的会话（即在同一个会话中用户提交的所有查询）和任务（相同目的的查询序列）。另一个问题是任务可以跨越多个会话，这种情况下任务称为研究任务。这个概念触发了雅虎的Search Pad。

另一种能够更精确判定会话的方法是设立最大不活跃时间。从Web导航日志中，不同的作者发现了不同的阈值，范围是5～60分钟［59，344，1479］。He和Gøker［632，728］在他们的日志样本上发现10～15分钟是最优的不活跃时间阈值。Huang等人［789］从信息论的角度考虑了从一个查询到另一个查询的变化，从而改进了阈值。他们在查询日志上的研究结果显示20分钟是一个较好的阈值，但是把阈值设为40分钟时得到了相似的结果。然而，Chen等人［364］指出阈值应该是依赖于用户和任务的，并提出了一种自适应的超时方法。近期，Jansen等人［825］调查了近250万个查询后发现，定义会话的最好方法是根据IP地址、浏览器的cookie信息，以及查询重构模式（即信息变化）定义。他们的研究结果表明93%的会话是由3个或更少的查询构成，平均每个会话有2.3个查询，这个结果与表7-2中显示的搜索引擎统计结果一致。

查询任务是一个重构查询的序列，重构查询表达的是相同的需求，通常是同一个信息需
273 求［847］。Radlinski和Joachims［1320］称这样的序列为查询链，而Baeza-Yates［90］称它为逻辑会话。任务检测是一个很难的问题，有些研究人员利用机器学习技术处理这个问题［847，218］。为了检测任务，重构的查询被定义为多种不同的类型：特化、泛化和拼写校正等。当任务发生变化时，重构查询被看做任务主题变化或者任务平行移动。首先定义这些查询类型的是［1356］，然后［218］扩展了类型，并利用查询流模型预测查询的重构形式［217］。

7.2.6 查询难度

查询的另一个重要特性是其内在自身的困难程度［336］。例如，单个词查询要比短语查询更加简单。因此，有些方法在文档集的上下文中评估查询的难度。当可以选择多个文档集时（参见10.3.2节），这是很重要的一点。

有两种不同的方法可以评估查询难度。最简单的一种方法就是运行查询并分析相应的答案集合。这种方法称为检索后预测机制。第二种方法更困难一点，就是在执行查询前评估查

询难度。这种方法称为检索前机制。

1. 检索后算法

检索后算法由于可以使用更多的信息，因此更加多样化。例如，清晰度（clarity score）[455] 根据集合的语言模型和得分最高的检出文档的语言模型之间的差异来预测查询难度。查询反馈 [1785] 考虑当使用排序最高的文档产生新的查询时发生的查询漂移。一种更激进的方法是文档和查询的扰动算法，即稍微改变查询项或者得分最高的返回文档，然后考察改变后产生的变化量 [1641]。Aslam 等人 [77] 提出了一种方法来评估由不同的检索算法产生的排序最高的文档之间的重叠是否相似。我们将在下面详细地讨论最主要的几种技术。

Cronen-Townsend 等人 [455] 提出了清晰度指标，该指标面向集合评估查询的歧义性。这个方法基于一种直观的想法，即无歧义查询排序最高的返回结果应该是与主题紧密结合的，并且与主题特别相关的索引项应该在那些文档中出现频率很高。另一方面，有歧义查询分布应该是与集合分布更加类似，因为排序最高的返回文档包含了各种各样的主题。举例来说，查询 "**artists who died in the 1700's**"[⊖] 是一个高难度的查询，因此很可能表现不佳。事实上，就像基于关键词的方法用关键词 "artist"，"die" 或者 "1700" 来检索包含这些关键词的文档集，这样的文档集将包含一组广泛的主题。清晰度的一种扩展是考虑查询的时间信息，这个方式是由 Diaz 等人提出的 [496]。 274

正如所有其他预测算法的性能，清晰度算法的性能依赖于文档集、检索设置，以及查询集。为了计算清晰度，对于一个给定查询，返回的排序文档列表用来建立查询的语言模型 [985]，其中那些在文档和查询中经常共现的索引项得到更高的概率。即

$$P_{qm}(k_i) = \sum_{d_j \in A} P(k_i|d_j)P(d_j|q)$$

其中 A 是检出的文档集，k_i 是词汇表中的单词，d_j 是文档，q 是查询。在查询模型中，$P(d_j|q)$ 可以使用贝叶斯公式估计：

$$P(d_j|q) = P(q|d_j)P(d_j)$$

这里，如果文档不包含查询项，那么其先验概率 $P(d_j)$ 为 0。由于对于所有的文档，概率 $P(q)$ 是相同的，因此省略了这一项。

典型地，概率估计需要通过重新分配某些概率块实现平滑，即赋给那些没有出现在查询中的项非零概率：

$$\begin{aligned} P(d_j|q) &= P(q|d_j)P(d_j) \\ &= P(d_j)\prod_{k_i \in q} P(k_i|d_j) \\ &\approx P(d_j)\prod_{k_i \in q \wedge d_j}(1-\lambda)P(k_i|d_j)\prod_{k_i \in q \wedge \neg d_j}\lambda P(k_i|C) \end{aligned}$$

这里 $P(k_i|C)$ 是根据文档集的语言模型 C 产生出的查询项 k_i 的概率，λ 是平滑参数（参见 3.5.2 节）。对于所有的查询项而言，参数 λ 是个常数，是在独立的测试文档集上的一个经验值。

清晰度定义了在查询语言模型 P_{qm} 和文档集语言模型 P_{coll} 之间的 Kullback-Leibler（KL）距离：

$$D_{KL}(P_{qm} \| P_{coll}) = \sum_{k_i \in V} P_{qm}(k_i)\log\left(\frac{P_{qm}(k_i)}{P_{coll}(k_i)}\right) \tag{7-1}$$

其中 V 是文档集上的词汇表，并且

⊖ TREC 查询 534 的标题。

$$P_{coll}(k_i) = \frac{F_i}{\sum_i F_i}$$

这里 $F_i = \sum_j f_{i,j}$ 含义如前（参见第3章）。KL距离越大，查询语言模型与文档集语言模型的差异越大。清晰度的唯一参数是排序靠前的文档数量（反馈文档的数量），这些文档用于采样生成查询语言模型。

Yom-Tov 等人 [1753] 比较了原始查询和查询成分项的排序列表。这种方法的思想是，
275 对于表现好的查询，如果只使用查询项的子集，那么结果列表也不应该发生很大的变化。他们应用机器学习方法，利用了各种特征，其中包括用原始查询和子查询获得的排序最高的文档之间的重叠、排序最高的文档的得分，以及查询项的数量。Aslam 等人 [77] 提出了基于相同思想的另一个方法：如果不同的排序方法检索返回的排序列表差异很大，那么这个查询被认为是难度大的。如果各个排序列表中排序靠前的文档的重叠很大，那么这个查询是比较容易的。为了评估性能，预测得分与从所有提交的 TREC 测试结果中的平均精度和中位精度进行了比较。

Zhou 和 Croft[1785] 研究了在 Web 搜索环境中估计查询难度的两种方法。*带权信息增益*（weighted information gain）度量了“从只有一篇普通文档被检出的虚构状态，到观察到真实搜索结果的实际状态之间的检索质量信息变化”（关于*信息增益*的详细讨论可以参见8.5节）。*查询反馈*将查询预测放在了通信信道问题的框架内考虑。输入是查询 Q，信道是检索系统，排序列表 L 是信道的噪声输出。根据排序列表 L，生成一个新查询 Q'，将 Q' 作为输入检出第二个排序列表 L'，计算 L 和 L' 之间的重叠作为预测得分。两个排序列表之间重叠得越少，查询漂移的可能性越大，查询的难度也越大。在 GOV2 上的实验表明这种方法取得了相当大的改进，超越了清晰度方法。查询反馈的参数是构成 Q' 的查询项数量 t，即 $t = |Q'|$，以及用于计算 L 和 L' 重叠的排序靠前的文档数量 s。

Hauff 等人 [717] 提供了一份对这些技术的综述和评估报告，同时也提出了*改进的清晰度*（improved clarity）。它与清晰度主要有两方面的区别。第一，反馈文档的数量可以自动设置；第二，索引项选择依据文档集中项出现的频率。他们证明在 Web 环境中，这种改进的清晰度要优于之前的评估方法。后来，Hauff 等人也提供了对这些技术的综述和评估 [715]，分析了什么时候应用这种方法是有效的 [714]。

2. 检索前算法

检索前算法必须依据文档集上查询项的统计来预测查询的难度。例如，既要考虑在文档集内查询项出现的频率，如*平均反比文档频率*（Averaged IDF）或*简化的清晰度*（Simplified Clarity Score），也要考虑文档集内查询项之间的共现频率，如*平均点态互信息*（Average Pointwise Mutual Information，PMI）。下面详细讨论这些主要技术。

Kwok 等人 [953] 提出了基于反比文档频率和文档集频率来区分弱查询和强查询，并使用了机器学习方法。*平均反比文档频率*是所有查询项的反比文档频率的平均值：

$$AvIDF(q) = \frac{1}{n_q}\sum_{k_i \in q}\log\left(\frac{N}{n_i}\right) \tag{7-2}$$

其中 q 是一个由 n_q 个索引项构成的查询，N 是文档总数，而 n_i 是包含索引项 k_i 的文档数量
276 （参见第3章）。通常认为项频低的查询能获得比项频高的查询更好的性能。

He 等人 [727] 评估了多种算法，包括*查询范围*（query scope）和*简化的清晰度*（simplified clarity score）。查询范围是根据文档集中至少包含一个查询项的文档数量来预测查询难度。简化的清晰度（Simplified Clarity Score，SCS）与平均反比文档频率的思想非常相

似，但这种方法用项频取代文档频率：

$$SCS(q) = \sum_{k_i \in q} P_{ml}(k_i|q) \times \log_2\left(\frac{P_{ml}(k_i|q)}{P_{coll}(k_i)}\right) \tag{7-3}$$

这里 $P_{ml}(k_i|q)$ 是在给定查询 q 的条件下，索引项 k_i 的最大似然估计。另外，$P_{coll}(k_i)$ 是文档集中索引项 k_i 出现的次数除以文档集中索引项的总数。

最后一种检索前算法是*平均点态互信息*（Averaged PMI），计算两个查询项在文档上的平均互信息，然后得到所有查询项对的平均值：

$$AvPMI(q) = \frac{1}{|(k_i,k_l)|}\sum_{(k_i,k_l)\in q} \log_2\left(\frac{P_{coll}(k_i,k_l)}{P_{coll}(k_i)P_{coll}(k_l)}\right) \tag{7-4}$$

这里 $P_{coll}(k_i,k_l)$ 是索引项 k_i 和 k_l 出现在同一篇文档中的概率。注意，对于单个索引项的查询而言，$AvPMI$ 为 0。

用检索前算法预测查询难度的研究工作经常产生混杂的结果，并且这些算法在准确率方面通常比检索后算法低。这是因为对这些算法有用的信息都更一般且更稀疏。尽管如此，最近提出的两种检索前算法取得了与检索后算法相当的性能。这两种方法都是计算密集型的，在 He 等人 [729] 的方法中需要用到聚类，而在 Zhao 等人 [1780] 的方法中需要计算所有文档的 TF-IDF 分布。尽管这两种方法是有效的，但是它们的效率并不足以让它们应用于大文档集，如 Web。

一个相关的问题是评价答案文档的质量。Vinay 等人 [1641] 考虑用多个预测器预测搜索结果的质量，包括一个基于检索返回文档聚类倾向的预测器，该预测器通过查看返回文档的"随机性"水平，从而确定聚类倾向。最近，Leskovec 等人 [1007] 提出一种监督学习方法，该方法从返回结果子图中抽取投影在整个 Web 图上的特征，建立准确推断返回结果质量的分类器，并且可以推断用户是否、以及如何重新生成查询。他们结论中的主要一点是，好的返回结果比差的返回结果连接得更紧密。近来，Barbosa 等人 [144] 提出了一种基于评估答案文档质量内聚力的技术，而不是基于评估查询难度的技术，结果显示查询的相关文档要比不相关文档有更高的内聚力，同时好的返回结果要比差的更加均匀。他们使用无监督学习技术，利用匹配的文档内容来推断搜索质量，简化并改进之前的研究工作。 277

7.3 趋势和研究问题

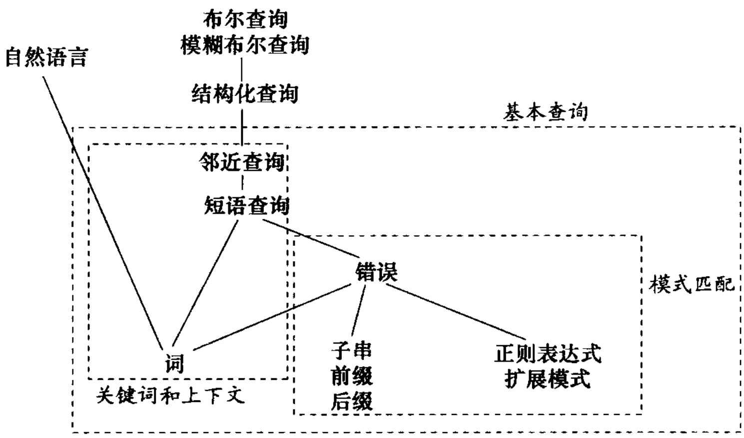
本章讨论了 Web 查询的主要特点以及用户的搜索行为。在这个主题下仍然还有许多没有解决的问题，包括更好地预测查询意图以及更好地用户建模。总之，挖掘查询现在已经是一个重要的研究领域，这部分将在 11.10.2 节中简要描述。

我们也讨论了从文本数据库中检索信息的查询语言的主要方面。我们涉及查询的方方面面，从最典型的方法到现在出现的最新颖的方法，从搜索单词到搜索扩展模式，从布尔模型到查询结构。表 7-4 列出了在不同的模型中允许的不同查询类型。尽管概率模型和贝叶斯信念网（Bayesian Belief Network，BBN）模型都是基于单词查询，但这些模型可以结合集合操作。

表 7-4 查询类型和模型之间的关系

模型	允许的查询
布尔	词、集合操作
向量	词
概率	词
贝叶斯信念网	词

在图 7-7 中，我们描绘了本章涉及的操作类型，以及它们之间的结构关系（并非所有的操作都存在于所有的模型中，也不是所有的操作必须用来表达查询）。这张图表明了我们可以在短语上使用布尔操作表达查询（跳过结构查询），也可以用单词或用正则表达式表达查询（跳过容错功能）。



278 图 7-7 涉及的查询类型以及它们之间的结构关系

全文数据库查询语言将朝着提供更灵活的查询描述发展。当文本模型正朝着更好地理解用户需求（如通过提供相关反馈）的方向发展时，查询语言在查询描述中也变得越来越强大。扩展模式和容错搜索能够在所需信息知识不完整的情况下发现那些模式，而且查询文本的结构也变得更普遍了。

另一个重要的研究主题是可视化查询语言。可视化表示可以帮助没有经验的用户生成复杂的布尔查询。另外，可视化查询语言可以包括文档结构［118］。这个主题是与用户界面和可视化相关的（参见第2章）。

一个重要的研究趋势是面向语义网的查询语言。现在查询 RDF 数据的标准语言是 SPARQL[1661]。RDF 是一个有标记的有向图，代表了 Web 上的信息（参见 6.4.4 节）。这种查询语言结合了 SQL 语言的能力以及额外的操作，以发现 RDF 数据中的图模式及它们的合取和析取关系。SQL 语言是关系数据库的标准查询语言。

7.4 文献讨论

典型的查询语言操作（最简单的模式、布尔模型和固定结构）可以在现在的商用系统中找到，如 Autonomy、Verity、Endeca、Fast 以及其他一些系统，也可以在早期的非商用系统中，如 Glimpse[1674] 或 Igrep[65]，以及更新的系统，如 Indri[809]、Lucene[1061]、Terrier[1574] 或 Zettair[1769] 中找到。

模糊布尔模型是在文献［1412］中介绍的。Levenshtein 距离是在文献［1012］和［63］中介绍的。Soundex 系统在［1913］中描述。文献［1195］中给出了不同相似度模型有效性的比较。一份很好的关于正则表达式介绍在文献［772］中。文献［1724］中介绍了一种扩展模式上的语言。

关于 Z39.50 系统的更多信息可以从文献［62］中获得。WAIS 系统的更多信息可以在文献［860］中看到。SFQL 的详细介绍可以参看文献［807］。

与 Web 相关的查询语言可以查看万维网联盟网站（World Wide Web Consortium, W3C）［190］。

想要获得更多 Web 查询的信息以及它们的特性，可以阅读［821，1517］这两本书。可惜，几乎没有关于其他信息检索系统的查询分析报告。文献［1714］是一篇关于 Web 会话分析的近期论文。

关于通常情况下的幂律信息可以阅读由 Newman 写的一篇很好的综述［1199］。

文献［716］是一篇很好的关于检索前预测查询难度的综述。近期更完整的综述是由
279 Carmel 和 Yom-Tov 完成的［335］。

文本分类

——与 Marcos Gonçalves 合著

8.1 介绍

早在公元前300年，亚历山大图书馆（Great Library of Alexandria）的馆员就已经开始对存储的文档进行处理，以便于日后进行检索与阅读。随着时间的推移，文档集越来越大，这一问题也就变得愈加困难。在数以百计的图书中顺序搜索来查找一本感兴趣的特定图书，已经变成了一个冗长、耗时且不现实的过程。为了缓解这一问题，图书馆员开始对文档进行标注。这样的举措可以为文档的内容提供元数据，因此就可以把图书按特定的视图进行组织，从而能够进行快速的查找与检索。

最初的文档标注方法是对每个文档赋予一个唯一的标识符。当用户知道他们所需要的图书时，这种方法可以很好地解决问题，但并不能处理更一般的问题——如何找到特定*主题*的文档。在这种情况下，一种很自然的解决方案是将文档按共同的主题进行分组，然后对每个组别赋予一个或者多个有实际意义的标签。

每个标注好的组称为一个*类别*（class），即是一个可以将内容与其标签描述相符的文档加入其中的集合。例如，我们可以构建一个标签为*心肌手术*的类别，并把所有描述治疗心肌疾病手术过程的文档都加入到这个类别中。在这个例子中，类别标签描述了文档中涉及的主题，因此这样的分类任务通常称为*主题分类*。这可能是最重要的一种分类问题，因此以下的讨论和例子都是基于该问题的。

类别不仅能够用来描述文档的主题，同时还能够表达与文档相关的其他特征，如语言、流派、质量、权威性、流行度以及垃圾信息等。例如，有一组对餐厅的评论意见，而我们需要区分出高质量的评论与低质量的评论。可以将这个问题建模为一个二元分类问题，两个类别为*高质量评论*和*低质量评论*。在这个例子当中，类别描述了文档（评论意见）的质量，而不是它们的主题或者评论意见的倾向性。 281

文档加入到类别的过程（即对文档赋予一个或多个类别标签）通常称为*文本分类*。有时，类别也称为*种类*（category），文本分类的英文"text classification"也写做"text categorization"。在本书中，我们把"classification"与"categorization"看做相同的过程而不加任何区分，都称为分类。

一个相关的问题是将一个文档集合分割成多个子集，但并不对其进行标注。由于每个子集都没有标签，因此不认为存在任何的类别。相应地，我们把每个子集称为一个*簇*（cluster），把子集分割的过程称为*文本聚类*（text clustering）。这里，我们把聚类当做一个更为简化的分类问题。

文本分类提供了一种组织信息的手段，以获得对数据更好的理解与解释。举例来说，有一家大型的工程公司多年来完成了许多大型项目。每个工程都有数以百计的设计文档、规划蓝图、档案信息、财务数据以及桥梁、隧道和铁路的资格证书。因此，产生了成千上万的关于该公司业务的文档。可以将所有文档分成多个类别，从而建立公司业务信息的结构化视图，对决策过程提供支持，因此具有很高的价值。这也说明文本分类已经成为现代企业进行

知识管理的关键技术。

本章将讨论关于文本分类的经典算法和评价方法，我们的讨论将包括监督算法和无监督算法。

8.2 文本分类的特性描述

许多文本分类算法都是由机器学习领域的研究人员设计出来的。因此，我们首先简要讨论机器学习与文本分类的关系。

8.2.1 机器学习

机器学习是人工智能［916，1106，1107，1122，1123，1399］范畴中一个广阔的领域，它主要研究算法的设计和开发，从而学习（learn）输入数据中所表达的模式［214，712，923，975，1140，1313，1446，1707］。这些经过学习得出的模式（可能是非常复杂的），可以用于预测那些未知的新数据。机器学习算法的应用包括自然语言处理、医疗诊断、信用卡诈骗检测、股票市场分析、计算机视觉和信息检索等领域。对于信息检索来说，文本分类是一个重要的问题，机器学习对它的影响很大。这一点我们会在本章的相关内容中详细阐述。

机器学习算法从根本上来说依赖于学习阶段，即产生一个能够对输入数据所表达的模式进行编码的模型或者函数。根据不同的学习机制，机器学习算法基本上可以分为三类：监督学习、无监督学习和半监督学习。其他学习算法包括了增强学习和直推学习，这里我们不对
282 它们进行讨论。

监督学习［214，712，923，975］需要从输入的训练数据中学习一个函数。在文本分类中，训练数据是由文档-类别对组成的，其中类别是由专家对给定文档标注的。这些训练数据可以用于训练分类函数，该函数对未知新数据的类别进行预测。这种方法只有在学习函数能够对未知数据做出高精度的预测时才会有效。监督学习是一些著名的文本分类算法的基础，我们将在8.4节具体地讨论这些方法。

无监督学习［147，916，1122，1123，1399］与监督学习的主要区别在于它没有训练数据。无监督学习算法包括了神经网络模型、独立成分分析以及聚类。对于文本分类的目标来说，聚类是其中我们最为感兴趣的无监督学习算法，我们将在8.3节进行讨论。

半监督学习［358］把少量的标注数据和大量的未标注数据结合起来，从而提高预测性能。而且预测性能的提高不需要耗费为大规模数据进行标注的时间。这里我们不对半监督学习做深入讨论，有兴趣的读者可以阅读参考文献［1787］，该文献对这一领域的研究工作进行了回顾。

监督学习和无监督学习对文本分类的算法设计会有直接的影响，这在随后的讨论中会表现得更为明确。

8.2.2 文本分类问题

文本分类问题可以形式化地表述为以下形式（我们的讨论受到了［1446］的很大影响）。

> **文本分类** 给定文档集$\mathcal{D}$与类别集合$\mathcal{C}=\{c_1,c_2,\cdots,c_L\}$，类别集合包含了$L$个类别及相应的标签。另外还给定了一个二值函数$\mathcal{F}:\mathcal{D}\times\mathcal{C}\rightarrow\{0,1\}$，即对每个文档-类别对$[d_j,c_p]$都赋予一个0或1的值，其中$d_j\in\mathcal{D}$，$c_p\in\mathcal{C}$。如果赋值为1，表示文档$d_j$是类别$c_p$的成员；如果赋值为0，表示文档$d_j$不是类别$c_p$的成员。

该文本分类的定义是宽泛的，既包含了监督算法又包含了无监督算法。为了说明无监督

算法的典型情况，考虑这样一个例子：在集合$\mathcal{C}$为空（即不提供类别）的情况下进行聚类。算法构建一组空的无标签的簇（簇的数目通常由输入确定），利用文档自身的属性来进行划分。然而在一般情况下，为了达到高精度的文本分类，我们应当采用监督算法。

如果对分类器没有特别的约束，那么同一个文档可能会被赋予两个甚至更多的类别标签。这种情况下，我们称这个分类器属于多标签（multi-label）类型。如果我们要求分类器对每个文档都赋予单一的标签，这个分类器属于单标签（single-label）类型。后面这种情况往往是更加困难的，因为我们不仅需要在每个文档-类别对上进行决策，而且还需要判断哪一个类别对于给定的文档 d_j 来说是最好的。这里，我们不涉及如何保证每个文档只属于一 283
个类别的额外步骤。相反，我们主要关注多标签的分类器。

上面所定义的分类函数$\mathcal{F}$是二值的，意味着判断文档 d_j 与类别 c_p 的成员关系是一个二元决策过程。但是，有时将函数$\mathcal{F}$构建为计算文档 d_j 对于类别 c_p 的隶属度。在这种情况下，对于每个类别 c_p 都有一组文档成为其候选成员。在信息检索中这是一种很自然的选择，排序函数将文档 d_j 的文本信息和类别 c_p 的标签信息作为输入并给每个文档-类别对赋予一个数值排序。而且我们可以利用这个排序对某个文档是否属于某个特定的类别做出决策。比如，我们可以在 $\mathcal{F}(d_j, c_p)$ 大于某个阈值时，认为文档 d_j 是类别 c_p 的成员；否则，就认为文档 d_j 不是类别 c_p 的成员。

8.2.3 文本分类算法

文本分类算法通常都是无监督算法或者监督算法。前者适用于大规模无标注的文档集，后者则更加复杂同时也能获得更好的结果，但需要可用的标注数据。

当训练样本不提供任何额外信息，即输入数据不包含文档属于哪个预定义的类别信息时，算法被称为是*无监督的*。图 8-1 显示本章将讨论的无监督算法。它们主要可以分为两类：聚类和朴素分类。对于聚类，我们讨论分割（partitioning）聚类与凝聚（agglomerative）聚类；对于朴素分类，我们将讨论一种基于向量模型的算法，这种方法将文档的索引项与类别标签进行直接的匹配。

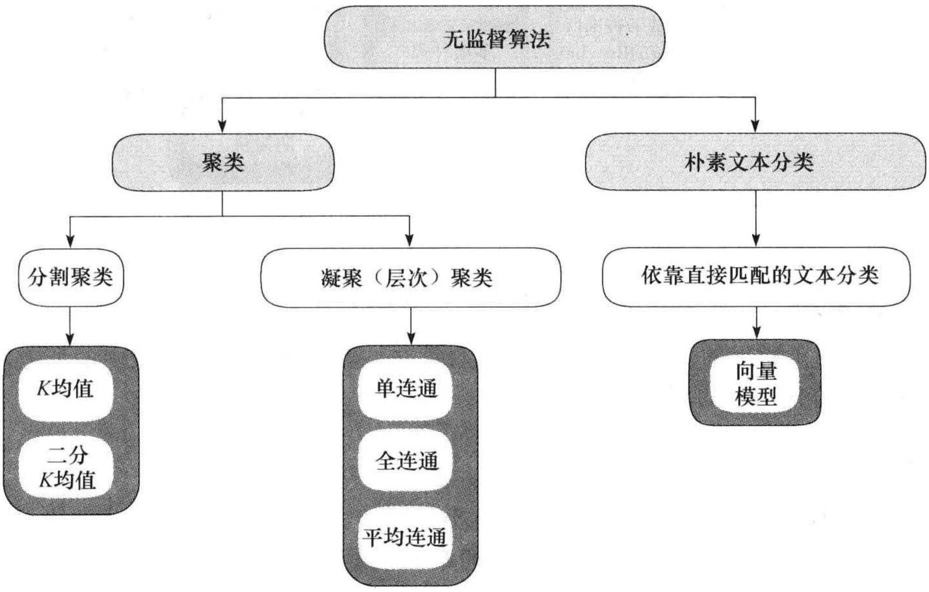


图 8-1 本章讨论的无监督算法 284

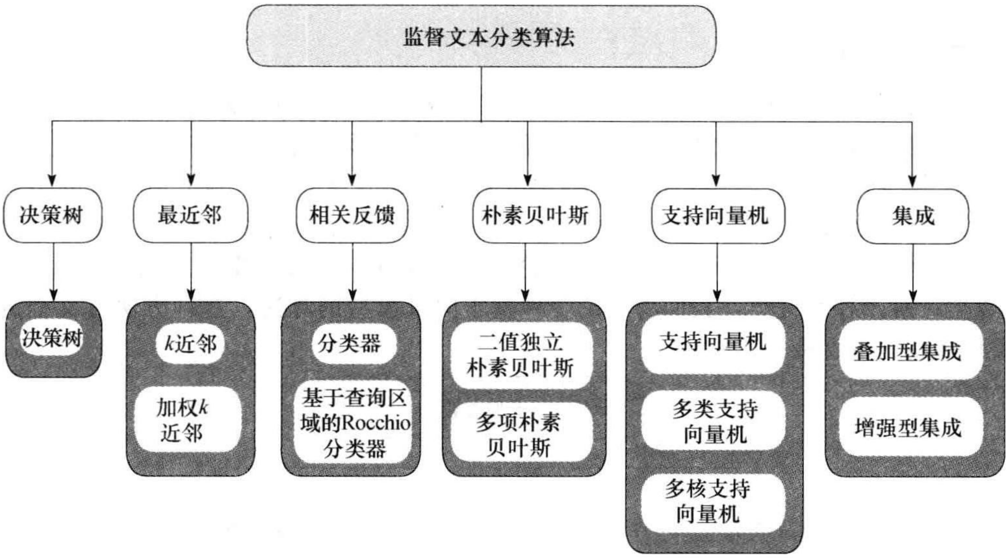
当一个算法使用了人工标注或者人工辅助标注信息时，这样的算法称为监督算法。在标准情况下，类别集合以及每个样本文档所对应的类别是给定的。这些由专家标注的样本组成了训练集（training set），可以用来学习分类函数。如果学习了该函数，那么就可以用它来对未知的新文档进行分类。

举例来说，有四个平均有 100 篇文档的类别，每篇文档都是由专家组进行类别标注。某个分类器可能会对所有的文档进行解析，然后找出每个类别中出现次数最多的索引项。这些索引项可以认为是这个类别的初始描述（description），从而用来对新的文档进行分类。

通常，训练样本的规模越大，分类器的效果就越好。然而，我们必须注意的一种情况是：分类器变得太特殊，可能只反映了训练样本的特征，而无法用来预测未知的新数据。这种现象我们一般称为过拟合（overfitting）。

为了对分类器进行评价，我们把它应用到一组已经事先确定类别的未知数据上，这样的数据集合我们称为测试集（test set）。如果分类器能够对测试集中大多数的数据做出正确的分类，那么我们就认为训练过程和得到的分类器是合适的。

图 8-2 显示了我们将在本章中进行讨论的六种文本分类的监督算法：决策树、最近邻、Rocchio 相关反馈、朴素贝叶斯、支持向量机以及集成学习。对于每个领域，我们将讨论其中最重要的算法和主要的变体。



285 图 8-2 本章讨论的监督算法

8.3 无监督算法

本节讨论图 8-1 所示的文本分类中的无监督算法。我们不会包括领域中的所有算法，而只是介绍其中具有代表性的算法。

8.3.1 聚类

假定训练数据只包含有文档集合，并没有类别标签。在这种情况下，分类器的任务是将文档分成多个组或簇（cluster），这个过程中通常也叫做聚类（clustering）。

文本聚类： 给定文档集D，文本聚类方法可以按事先定义的准则将这些文档自动地分为

K 个簇。

图 8-3a 显示了将夏威夷的酒店的网页分成五个簇的过程，每个簇都是由同一个岛屿上的酒店组成的（即这个例子中的聚类标准是地理邻近性，更为靠近的酒店会被分到同一个簇中）。尽管这一聚类过程对人类来说是很自然的，但却很难由完全自动化的步骤来实现。其中的原因在于酒店的网页中有许多索引项都是相同的，没有人类的帮助理解，是很难辨识上下文中的哪些索引项对描述酒店位置有较大影响。因此，聚类方法面临着这样一个固有的问题：自动生成的簇常常与人类的直觉结果不相符合。

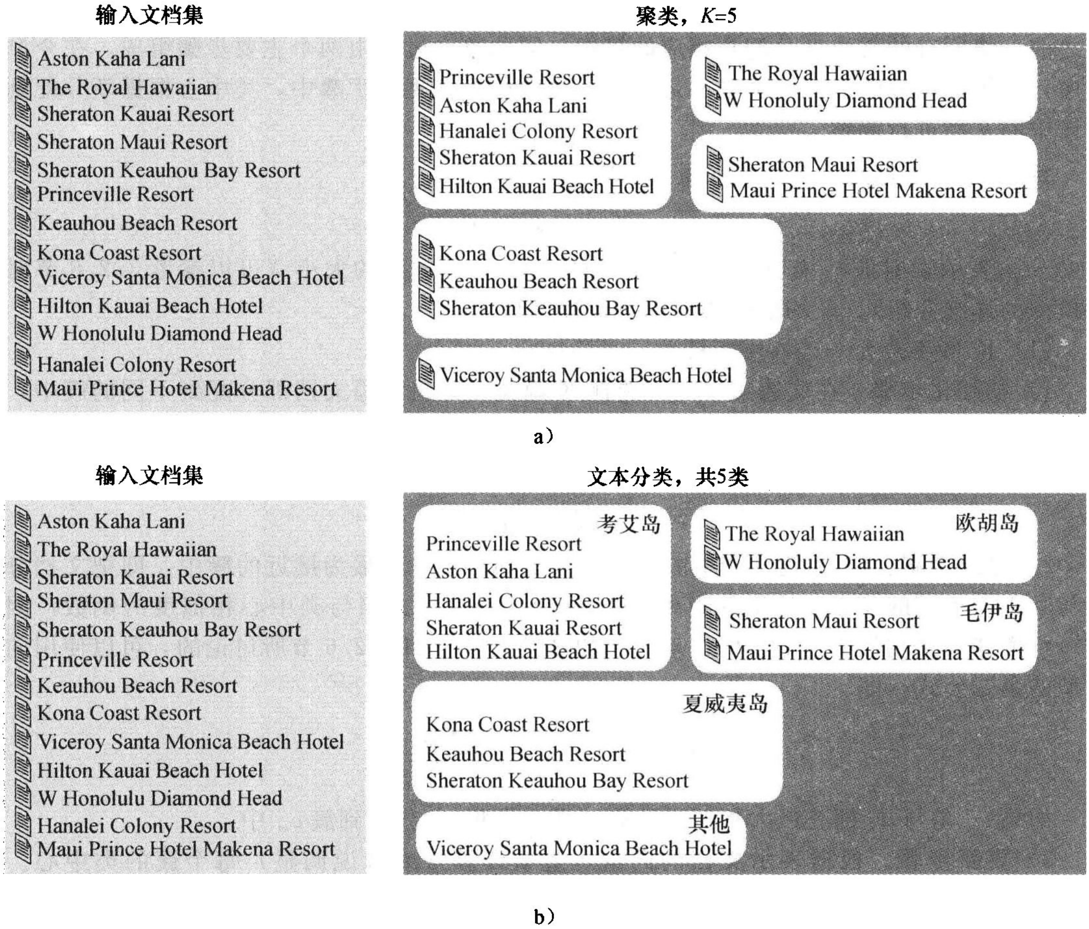


图 8-3 无监督文本分类方法应用于夏威夷酒店的网页：a）聚类的结果；b）酒店所分配的类别

倘若要求对聚类的结果进行标注，那么这个问题就会变得更加困难。依然以图 8-3a 为例，这个任务进一步要求我们标识每个簇中酒店所在的岛屿。如果完成顺利的话，结果将如图 8-3b 所示。十分遗憾的是，自动生成的标签往往与图示的情况有很大的差别，并且对人来说没有明确的含义。基本上，自动生成簇标签仍旧是一个非常困难的问题。

尽管产生的结果很难解释，但聚类还是能够帮助我们更深刻地理解数据并且发现数据的一些自然属性。一个例子是处理每日数以千计的银河系图片。通常并不存在明确的先验方法来对数据进行组织。因此，一种可能的过程是将它们按共同的特征进行聚类，然后从最紧密的簇当中寻找数据的模式。同样的方法还可以用于分析人类语言、股票和基因序列等。

为了说明文档聚类的一般过程，我们现在来讨论一种著名的分割算法——*K* 均值（*K*-

means）以及其变体二分 K 均值。我们同样会讨论三种凝聚聚类的方法，包括单连通、全连通和平均连通。

1. **K 均值聚类**

在 K 均值聚类中［1069］，簇的数目 K 通常是由输入指定的。每个簇由一个中心点来表
286 示，称为类中心（centroid）。文档集被划分为 K 个簇，将每篇文档分配到与类中心最为接近的簇中。当所有的文档都划分完之后，每个簇的类中心都会被重新计算。整个过程会不断地重复，直到类中心不再改变。

基本的 K 均值方法是按批处理模式（batch mode）进行的［573］，即在重新计算类中心之前，将所有的文档都分配到相应的簇中。因此整个过程由两个主要步骤组成。在*分配步骤*中，每篇文档被分配到与类中心最为接近的簇中。在*更新步骤*中，类中心根据新分配到这个簇中的文档进行调整。

每篇文档 d_j 都用一个权重向量 $\vec{d}_j$ 表示，即有

$$\vec{d}_j = (w_{1,j}, w_{2,j}, \cdots, w_{t,j})$$

其中 $w_{i,j}$ 表示索引项 k_i 在文档 d_j 中所占的权重，t 是词典的大小（可以参考 3.2.6 节对向
287 量模型的详细介绍）。K 均值聚类按以下过程进行。

（1）*K 均值聚类——批处理模式*

1）**初始化步骤**。在文档集中随机选择 K 篇文档，将每篇文档都分配到不同的簇中，而这些文档就作为初始的类中心。举例来说，如果 d_j 是初始选中的某篇文档，c_p 是其所属的簇，$\vec{\Delta}_p$ 为相应的类中心，则有

$$\vec{\Delta}_p = \vec{d}_j$$

2）**分配步骤**。将 N 篇文档中的每篇都分配到与类中心最为接近的簇中，即把文档分配到与其距离最短的簇中。这里的距离（distance）表示为文档与类中心相似度的倒数，因此最短距离和最大相似度是等价的。对于相似度计算，正如 3.2.6 节所讨论的，可以使用向量模型的余弦公式，即

$$sim(d_j, c_p) = \frac{\vec{\Delta}_p \cdot \vec{d}_j}{|\vec{\Delta}_p| \times |\vec{d}_j|}$$

如果 c_m 是与 d_j 拥有最大相似度的簇，那么便把 d_j 分配到簇 c_m 中。

3）**更新步骤**。根据表示文档的向量，重新计算（或者说调整）每个簇的类中心。令 $size(c_p)$ 是簇 c_p 中所包含的文档数量。那么，相应的类中心 $\vec{\Delta}_p$ 可以按以下的形式重新计算：

$$\vec{\Delta}_p = \frac{1}{size(c_p)} \sum_{\vec{d}_j \in c_p} \vec{d}_j$$

4）**最终步骤**。重复步骤 2 和步骤 3 直到类中心不再发生改变。

K 均值算法的第 2 个版本是以在线（online）形式操作的［1532］，即类中心在每个文档进行划分后都进行重新计算。它的操作形式如下。

（2）*K 均值聚类——在线模式*

1）**初始化步骤**。在文档集中随机选择 K 篇文档并以这些文档作为初始的类中心。

2）**分配步骤**。对于每篇文档 d_j 重复以下过程

- 将文档 d_j 分配到与类中心最为接近的簇中。
- 重新计算这个的类中心。

3）**最终步骤**。重复步骤 2 直到类中心不再发生改变。

在参考文献［1532］中，作者提出在一般文本集上，在线形式的 K 均值算法比批处理形式的 K 均值算法获得更好的结果。

K 均值聚类可能在一些情形下工作得很好，但在另一些情况下却不能取得理想的结果。如何选择簇的数目 K，对于算法来说是至关重要的一个步骤。另外，受随机选择的初始类中心点的影响，多次运行这一算法可能会得到不同的聚类结果。 288

2. 二分 K 均值算法

这一算法构建了一个层次化的聚类结构，每一次都把某个簇分割成两个簇。而这一过程是通过反复应用 K 均值方法（$K=2$）完成的。

二分 K 均值算法的步骤

1）**初始化步骤**。将所有文档分配到同一个簇。

2）**分割步骤**。对于最大的一个簇，应用 K 均值算法（$K=2$）。

3）**选择步骤**。如果满足诸如“没有任何簇比某个给定的阈值要大”这样的终止条件时，算法停止执行；否则，选择文档数目最大的簇应用步骤2）。

K 均值算法还有其他变体形式，比如期望最大化算法（Expectation Maximization，EM）。其主要的思想是对聚类结果使用基于概率的分配，而不是确定性的。这些聚类算法已经超出了本章的范围，有兴趣的读者可以参考文献［817］。

3. 层次式凝聚聚类

相对于 K 均值算法将集合划分成 K 个簇，层次聚类方法构建了层次化的聚类结构。这类方法要么将大的簇分解成小的簇，要么将预先定义的簇合并成大的簇［817］。一般的文档层次式（凝聚）聚类算法可以描述为以下的形式：

层次聚类

1）**步骤1**。以一个 N 篇文档的集合和一个 $N \cdot N$ 的相似矩阵（或距离矩阵）为输入。这个矩阵中的数据项可以是如向量模型的余弦函数值一类的结果。

2）**步骤2**。把每个文档都分配到不同的簇中，每个簇包含一篇文档，这样就产生了 N 个簇。这个簇就代表了树中的一个叶子。任意两个簇的相似度（或者距离）就是其包含文档的相似度（或者距离）。

3）**步骤3**。找出最为相似（或者最接近的）的一对簇，并把它们合并为一个簇，同时簇的数目减1。这个新簇可以用树中比原有簇高一层次的结点来表示，同时作为原有簇结点的父结点。

4）**步骤4**。利用一个定义在文档集上的函数，重新计算新簇和其他簇的相似度（距离）。

5）**步骤5**。重复步骤3和步骤4，直到所有文档都被合并到一个簇中，其大小为 N。中间的聚类结果以及初始的单文档簇共同组成一棵树，即层次化的聚类结果。 289

步骤4需要计算两个簇的相似度（或距离）。计算簇的相似度（或者距离）有三种不同的方法，分别为单连通（single-link）、全连通（complete-link）和平均连通（average-link）。

- **单连通算法**。簇之间的距离定义为分别属于不同簇的任意两篇文档之间距离的最小值（或相似度的最大值）。
- **全连通算法**。簇之间的距离定义为分别属于不同簇的任意两篇文档之间距离的最大值（或相似度的最小值）。
- **平均连通算法**。簇之间的距离定义为分别属于不同簇的任意两篇文档之间距离的平均值。

有一类称为基于密度的聚类方法，如 DB-SCAN[537]，其方法除了考虑距离因素之外，还会考虑文档空间中邻近区域（可以是任意形状的）内邻居点的个数。对于这类方法的进一步讨论已经超出本章的范围。

8.3.2 朴素文本分类

另一种形式的无监督分类不需要依靠训练样本的信息来定义类别，具体如下所示。

> **朴素分类** 给定文档集$\mathcal{D}$与类别集合 $\mathcal{C}=\{c_1,c_2,\cdots,c_L\}$，类别集合包含了 L 个类别与相应的标签。在没有训练数据的情况下，使用一种方法可以自动将集合中的文档关联到一个或者多个类别$\mathcal{C}$。这种方法也常常称为分类器。

由于类别标签是可用的，因此一种简单的分类算法实现是直接匹配文档的索引项与类别标签。为了提高分类算法的覆盖率，我们可以对每个类别定义其替代标签。这种标签通常称为同义词（synonym），尽管它们可能并不保持语法上的同义关系。通过我们即将讨论的 TF-IDF 权重以及在向量模型中余弦公式的应用，可以量化部分匹配。

依靠直接匹配的文本分类

1）**步骤 1**。以文档集$\mathcal{D}$与类别集合 $\mathcal{C}=\{c_1,c_2,\cdots,c_L\}$ 为输入，类别集合包含了 L 个类别与相应的标签。

2）**步骤 2**。将文档和类别表示成向量模型中的索引项权重向量，从而文档 d_j 表示为向量 $\vec{d}_j$，类别 c_p 表示为向量 $\vec{c}_p$。类别向量是通过组成类别标签的索引项构建的。

290 3）**步骤 3**。对于每一篇文档 $d_j\in\mathcal{D}$，

- 检索出类别 $c_p\in\mathcal{C}$，其标签与文档 d_j 中的索引项匹配。
- 对于每对 $[d_j,c_p]$，通过以下公式计算其基于向量模型的排序

$$sim(d_j,c_p)=\frac{\vec{d}_j\cdot\vec{c}_p}{|\vec{d}_j|\times|\vec{c}_p|}$$

- 将文档 d_j 与具有最高 $sim(d_j,c_p)$ 值的类别 c_p 关联。

为了改善结果，文档向量和类别向量可以用所有索引项的一个子集表示为特征向量的形式，具体可以查看 8.5 节。

尽管非常简单，但朴素文本分类对于集中于某个特定领域知识的垂直文本集还是能产生良好的效果 [1032，1351]。当有明确的分类体系，即类别可以按特化/泛化关系进行层次化组织时，这一结果特别明显。然而，对于一般的文档集，由于匹配文本索引项与类别标签的方法可能有很大的局限性，因此朴素文本分类会产生糟糕的结果。在一般情况下，为了提高分类的效果，我们有必要采用监督算法。在 8.4 节将进行讨论。

8.4 监督算法

通过增加预先由人工分类的样本文档来训练（或微调）分类器，文本分类问题可以通过更加精细的方法进行处理。

1. 分类器训练

在应用于大规模文档集之前分类器通常需要进行精细的调整，而这种调整是基于训练集（training set）的，训练集的定义如下。

> **训练集** 给定 $\mathcal{D}_t\subset\mathcal{D}$ 是文档集的子集，训练集函数 $\mathcal{T}:\mathcal{D}_t\times\mathcal{C}\rightarrow\{0,1\}$，根据专家的判断，给每对 $[d_j,c_p]$ 赋予一个 0 或 1 的值，其中 $d_j\in\mathcal{D}_t$，$c_p\in\mathcal{C}$。

训练集函数 $\mathcal{T}$ 用来对分类器进行精细的调整，提高分类函数 $\mathcal{F}:\mathcal{D}\times\mathcal{C}\rightarrow\{0,1\}$ 的精度，如图 8-4 所示。最终得到的分类函数$\mathcal{F}$应当具有一定的泛化特性，可以用来预测那些未知的新文档。

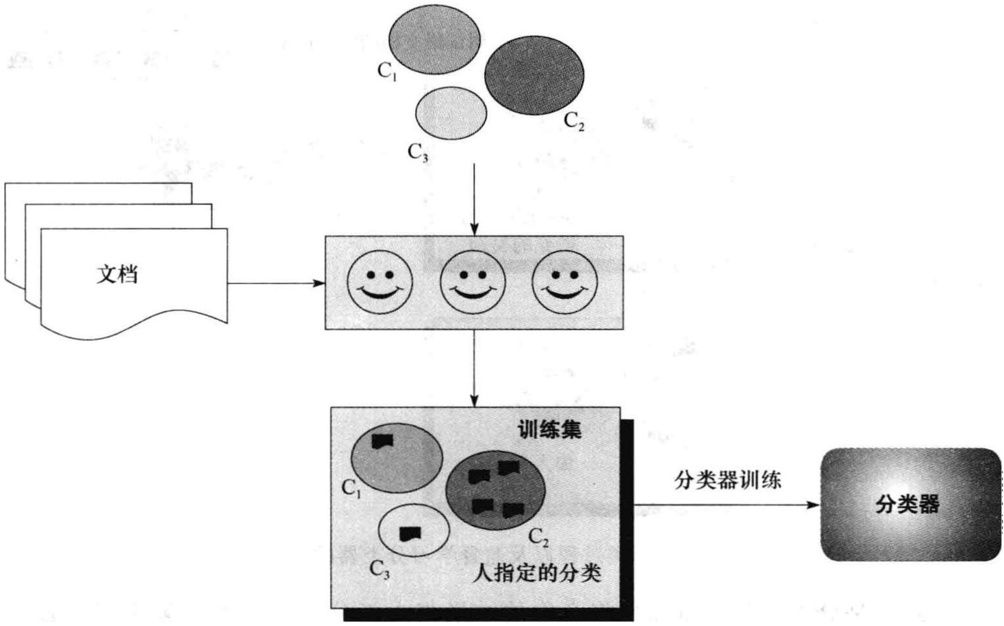


图 8-4 分类器的训练步骤

2. **分类器评价**

为了对分类器进行评价，我们预先选择一组类别已知的文档——称为测试集（test data)。与训练集相同，测试集中文档所属的类别是由专家进行判断的。而且，测试集与训练集应当是不相交的，从而保证最终的评价是基于未知的新文档。

将测试集用于分类器的评价，分为两个步骤：1）使用分类器将测试集中的文档进行分 291
类；2）将分类器赋予的类别与专家标出的类别进行比较。整个过程如图 8-5 所示。给定测试集中的文档后，我们首先将其进行索引，得到合适的文档表示。一种可能的表示方式是使用文档的全文视图（full text view)，即文档 d_j 是由其所有索引项所组成的集合来表示的。另一种表示方式称为部分文本视图（partial text view)，通过词干提取和禁用词移除等方式，文档 d_j 可以由其所有索引项的一个子集来表示。无论哪种表示方式，全文或者部分，都应该给每个索引项包含类似 TF-IDF 的权重，就像我们在 3.2.4 节讨论的那样。

为了达到分类目的，文档表示中的每一个索引项都被视做一个独立的变量，我们称之为特征（feature)。这意味着文档表示的大小可能达到文档本身大小的级别，也就可能导致特征空间维度会变得非常庞大。对于高维的特征空间，更复杂的方法就难以使用，因为这会导致计算复杂度变得非常高。为了缓解这一问题，可以选择索引项的一个子集来表示文档。这一过程称为特征选择（feature selection)，我们将在 8.5 节讨论。这个过程可以将文档表示约化为特征向量（feature vector)，然后再提交给分类器。

评价分类器的最后一步是比较分类器产生的结果与专家产生的结果。通常由专家提供的结果不会完全等同于自动分类的结果。尽管如此，自动产生的结果与专家提供的结果一致率越高，分类器的效果也就越好。我们将在 8.6 节具体讨论如何量化这种效果。 292

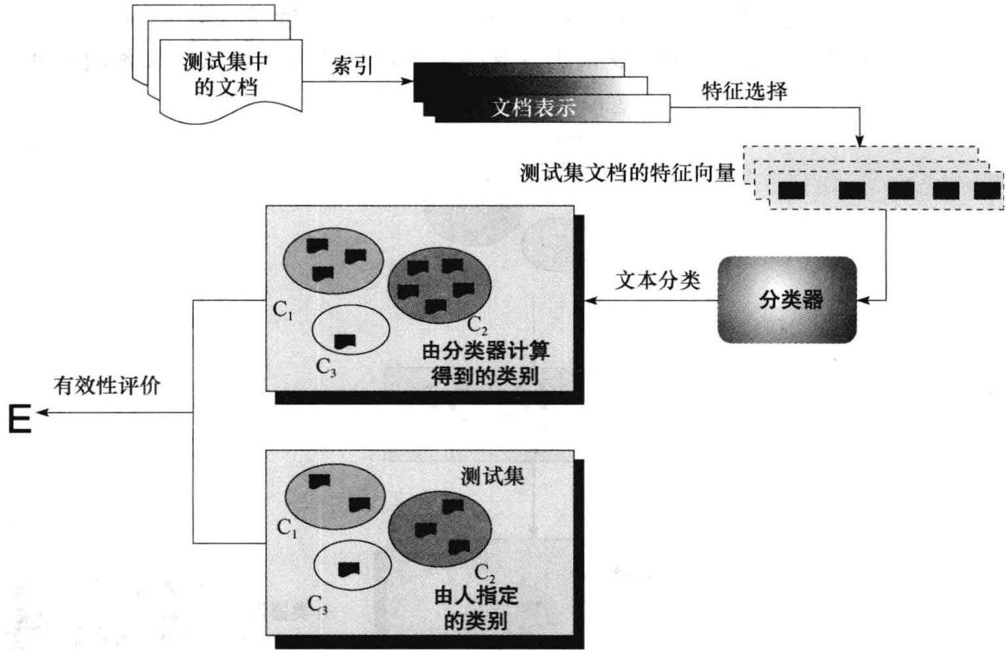


图 8-5 分类过程以及监督学习分类器的评价

需要注意的是分类器的评价需要将产生的结果与人工分类产生的结果进行比较。然而，人们常常在一篇给定文档的最佳分类问题上产生分歧，即人工分类由主观因素主导的。举例来说，考虑一篇文档讨论了在18世纪下半叶，瓦特（James Watt）发明的蒸汽机对于当时经济的影响。有些人可能会将这篇文档分类到“机械工程”或者“蒸汽机”的类别当中，而其他人可能将其分类到“18世纪经济”或者“工业革命”的类别中。也有可能把这篇文档同时分类到所有以上的类别中。这也说明文本分类是一个受主观解释影响的问题，文本分类器能达到的最好结果是近似人工的分类过程。

3. **文本分类**

分类器在经过了训练和验证后，就可以用于对未知的新文档进行分类。这可以按图8-5所示的步骤进行，只是不包括与人类产生的结果进行比较的最后一步。当然，在这种情况下，作为输入的文档应当不包含在测试集中。如果分类器工作正常的话，我们希望可以有效地将新文档分类到正确的类别中，即大部分的类别预测应当是正确的。

接下来，我们将讨论如图8-2所示的文本分类监督算法。我们并不试图覆盖文献中的所有分类算法，而只是覆盖那些有代表性的方法，其中许多方法都可以使分类器达到最好的性
293 能表现。对于每个算法，我们会介绍其使用的基本方法作为基础。我们也会讨论如何改变方法使其能够用于文本分类的问题，同时在适当的时候提供一些应用的例子。

8.4.1 决策树

决策树（Decision Tree，DT）属于监督分类方法，使用训练集从而将分类规则组织成一棵树中的路径。这些树的路径可以用来对训练集以外的文档进行分类。这种方法的一个优点在于树中的规则更符合人类的理解方式，而不像其他方法，比如朴素贝叶斯（Naïve Bayes）或者支持向量机（Support Vector Machines）。这是因为决策树提供了显式的数据结构（数据路径树）可以使得分类过程的结果更容易被理解。

1. 基本技术

在如表 8-1 所示的小型关系数据库中，其模式是由 Id、Play、Outlook、Temperature、Humidity 和 Windy 等属性组成的。每个元组都是由像 "Id=5" 和 "Outlook=rainy" 一类的属性值组成的。 294

表 8-1 训练与测试实例

	Id	Play	Outlook	Temperature	Humidity	Windy
训练集	1	yes	rainy	cool	normal	false
	2	no	rainy	cool	normal	true
	3	yes	overcast	hot	high	false
	4	no	sunny	mild	high	false
	5	yes	rainy	cool	normal	false
	6	yes	sunny	cool	normal	false
	7	yes	rainy	cool	normal	false
	8	yes	sunny	hot	normal	false
	9	yes	overcast	mild	high	true
	10	no	sunny	mild	high	true
测试实例	11	?	sunny	cool	high	false

用于该数据库的决策树构建了一个数据结构，可以预测某个给定属性的值。举例来说，图 8-6 中的决策树（来自参考文献 [1312]）可以在给定 Outlook、Temperature 和 Windy 等属性的条件下，预测属性 Play 的结果。在决策树中，内部结点（非叶子）与属性名相关联而边则与属性值相关联。

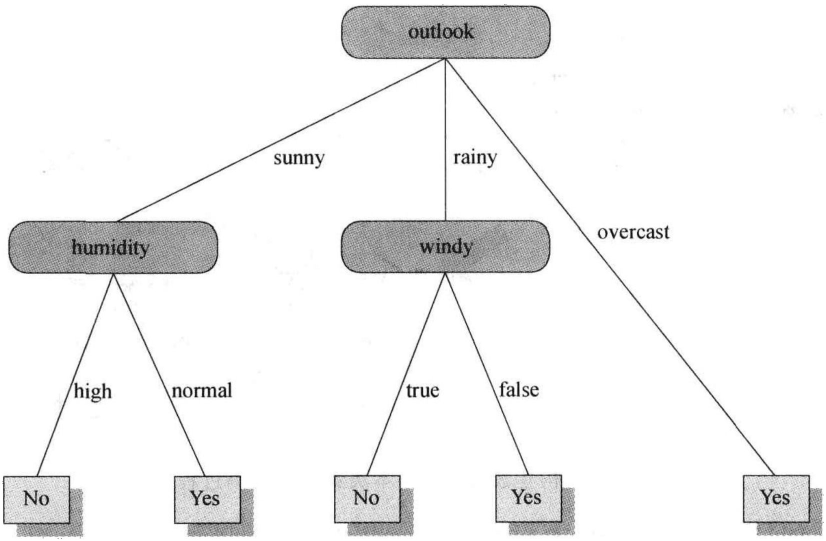


图 8-6 由表 8-1 的训练集所导出的决策树

当决策树需要对一个新的实例（如表 8-1 中的元组 11）进行分类时，基于该实例的属性值对这棵树进行递归遍历，从而决定属性 "Play" 最为合适的属性值。在这个例子中，根据规则路径 $(outlook = sunny) \wedge (Humidity = high)$，结果最终为 "not to play"。

需要注意的是，我们的预测都是基于样本数据库中已知的实例。如果一个新的实例与数

据库中原本包含的隐式规则不符，那么将会导致预测错误。对于决策树的构建来说，样本数据库就是其训练集。具体方法采用了分割的过程，如下所示。

2. 分割过程

给定训练集（数据库），决策树模型可以通过递归分割策略来构建，具体过程如下：假设目标是构建能够预测属性 Play 值的决策树。第一步我们选择除 Play 以外的某个属性，以它为决策树的根，相应的属性值就用来将数据库的元组分割成若干个子集。对于每一个元组子集，再选择一个属性进行分割，然后不断重复这一过程，直到每个子集中的元组都只包含相同的 Play 属性值。

图 8-7 按步骤显示了分割过程。第一个被选中的分割属性是 Outlook，其属性值将元组分割成 3 组：{4，6，8，10}、{1，2，5，7}、{3，9}。对于第三组来说，所有元组都有相同的 Play 属性值“yes”。这也就意味着不需要进一步分割了。根据训练数据，当“Outlook＝overcast”时，有“Play＝yes”。其他两个元组子集需要进一步分割。对于第一个元组子集，选择用来分割的属性是“Humidity”。对于第二个元组子集，选择用来分割的属性是“Windy”。在第三次分割之后，针对样本数据库的决策树就构建完毕了。

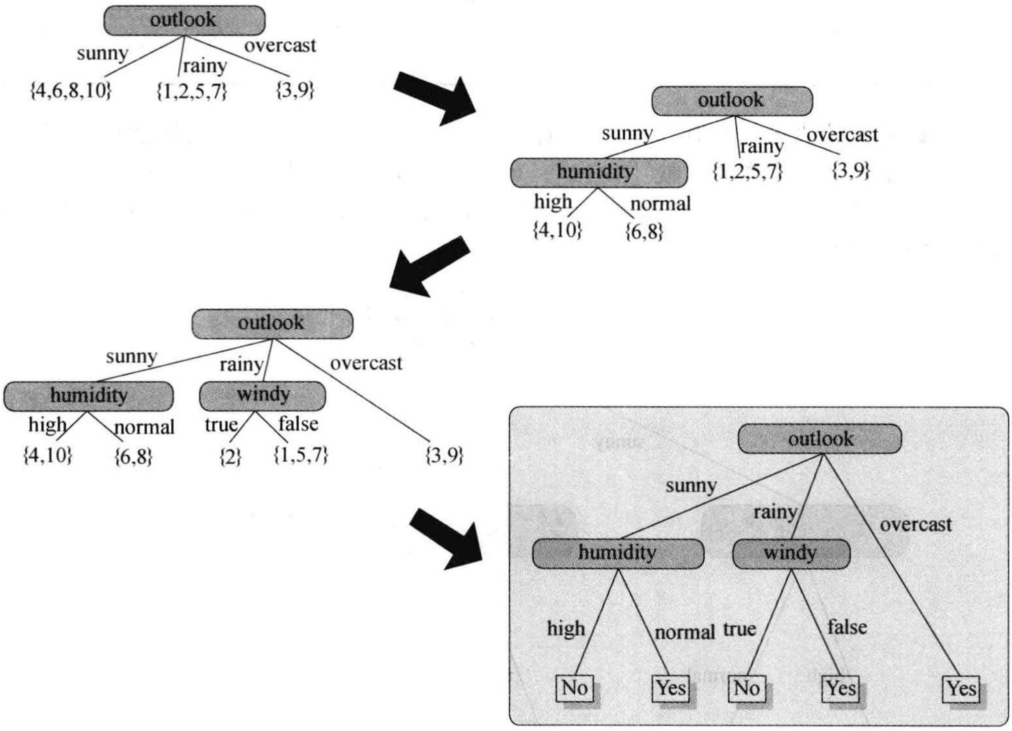


图 8-7 构建表 8-1 中样本决策树的分割过程

需要注意的是，分割过程很大程度上受到用来分割的属性顺序的影响。由于顺序不同，树可能会变得不平衡，即从根到叶子的不同路径可能差别很大。这是我们在设计分
295 割策略时面临的主要挑战。通常平衡或者近似平衡的树在预测属性值时有更好的效果，同时也更为高效地完成各种操作。因此，一种普遍的经验方法是选择那些能够减少根到所有叶子平均路径长度的属性。对于图 8-6 中的决策树来说，根到叶子的最长距离是 2，最短距离是 1。

3. 文档分类

对于文档分类，树中的每一个中间结点都关联到一个索引项，即每个索引项都认为是与文档关联的独立变量。树中的每个叶子都关联到一个文档类别，而每条边都表示某个索引项出现与否的二值预测。

(1) 分割过程

用于文档分类的决策树构建会采用与之前类似的分割过程。下面，我们形式化地明确一些基本概念。

定义 V 为结点集合。树 $T=(V,E,r)$ 是 V 上的有向无环图，其中 $E\subseteq V\times V$ 为边集合，$r\in V$ 为 T 的根。给定边 (v_i,v_j)，v_i 为父结点，而 v_j 为子结点。根是唯一没有父结点的 296
结点。结点 v_k 称为叶子，当且仅当其没有子结点。我们指定 I 为所有中间结点（非叶子）的集合，而 $\overline{I}$ 为所有非中间结点（即叶子）的集合。

给定树 T，我们可以将训练集中的文档与类别的信息关联到树上。依靠这样的方法，我们能够构建用于文档分类的决策树，具体如下所示。

定义 设 $K=\{k_1,k_2,\cdots\}$ 为训练集 $\mathcal{D}_t$ 所有索引项的集合，而 C 是所有类别的集合，如前所示。而且，设 P 是基于索引项的逻辑谓词的集合。决策树 $DT=(V,E;r;l_I,l_L,l_E)$ 为一个六元组，其中 $(V;E;r)$ 是一棵根为 r 的树；$l_I:I\rightarrow K$ 是将树上的中间结点关联到一个或者多个索引项上的函数；$l_L:\overline{I}\rightarrow C$ 是将非中间结点（叶子结点）关联到某个类别 $c_p\in C$ 上的函数；$l_E:E\rightarrow P$ 是将树的边关联到 P 中的逻辑谓词的函数。

给定由文档及其相应所属类别组成的训练集，决策树模型可以按照类似于图 8-7 所示的方法，通过递归分割策略 [1140，1446] 来构建。第一步将所有文档关联到根结点。第二步选择一部分能够将与根结点关联的文档进行良好分割的索引项。举例来说，假设 k_a、k_b、k_c 和 k_h 用于第一次分割，于是根结点中的文档就被分为 4 个子集，每个子集都关联到树中的一个新的子结点，在与子结点相连的边上标记相应索引项的谓词，如 $P(k_a)$。为了满足该谓词，某篇文档 d_j 需要匹配其条件。譬如，谓词可以由两个待满足条件组成：1）文档 d_j 需包含索引项 k_a；2）关联到索引项-文档对 $[k_a,d_j]$ 的权重 $w_{a,j}$ 应当超过某个给定的阈值。这可以表示为：

$$P(k_a)=contains(k_a,d_j)\wedge w_{a,j}\geqslant\tau$$

其中 τ 是满足该谓词的最小阈值。权重可以通过 3.2.6 节中讨论的经典向量模型中的 TF-IDF 公式计算得到。文档 d_j 满足谓词 $P(k_a)$，当且仅当其包含索引项 k_a 且与 $[k_a,d_j]$ 关联的权重超过 τ。需要注意的是，阈值 τ 可能随着索引项的变化而变化（即不一定在所有索引项中都取相同的常数值）。接下来，每个文档子集都选择新的分割项，以上的过程不断重复递归进行。在每个分支，递归过程在子集的所有文档都归属于同一个类别时终止。

图 8-8 显示了一个例子。在第一步，选中索引项 k_a、k_b、k_c 和 k_h，并将集合分为 4 个文档子集。子集 $\{d_4,d_{10}\}$ 和 $\{d_5,d_6\}$ 仅包含分别属于类别 C_a 和类别 C_f 的文档，因此对于这两个分支不需要进一步地分割。接下来，选择索引项 k_d、k_e、k_f 和 k_g 用来进行分割并重复上面的过程。在分割过程中，将中间结点插入决策树。我们对这些结点进行简单的编号，标记并无任何特殊含义。我们观察到在一个大的文档集中，决策树的规模也会变得庞大而构建过 297
程也会耗费更多的时间。

选择分割项的过程是最为重要的。尽管可以采用不同的方法，但是互信息（熵）和信息增益（参考 8.5 节）等是最为常用的方法。选择能够带来高信息增益的索引项可以增加某个

给定层次的分支数量，从而减少每个子集中的文档数。这样可以产生规模更小、结构更简单的决策树。

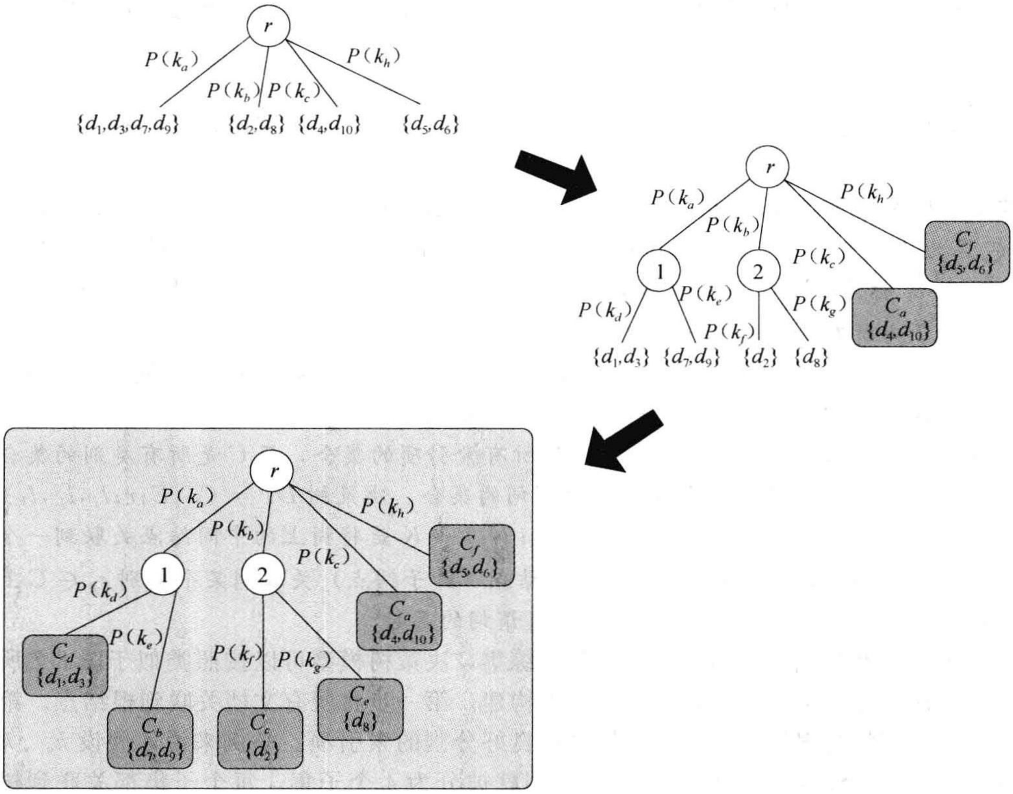


图 8-8 根据文档集构建决策树的分割过程

(2) 分类新文档

为了对一个新文档进行分类，我们通过对文本的索引项和树中边所指代的索引项进行匹配的方式，对决策树进行遍历。在遍历到树的某个层次时，如果文档满足边上的谓词，那么它就包含在其相关联的那个子集中。需要注意的是，同一篇文档可能会关联到两个或者多个文档子集中。递归遍历的过程不断重复，直到达到某个叶子结点。与这个叶子结点相关联的类别就是最终的文档类别。

决策树存在一些固有的问题，如缺失值或者未知值 [595]。这一类现象出现在待分类文档不包含决策树中分割项的时候。这种情况下，我们不容易判断遍历过程应该选择树的哪一个分支。解决这一问题的一种方法是延迟树的构造，直到待分类的新文
298 档可用为止。然后，树只依靠这篇文档本身具有的特征来进行构造，从而避免了上面的问题。这种"延迟"技术同样被接下来要介绍的 k 近邻分类器 (k-Nearest Neighbor, kNN) 所使用。

8.4.2 k 近邻分类器

k 近邻分类器 (k-Nearest Neighbor, kNN) 是一种面向需求的 (或者延迟的) 分类器。延迟分类器并不预先构造一个分类模型。相反，分类过程只在新文档 d_j 输入分类器时才进行。分类的决策是基于文档 d_j 的 k 个"最近"的邻居所属的类别来判断的，这些邻居通过

在预先定义的度量空间上的距离函数来计算。具体过程如下：1）确定文档 d_j 在给定训练集中最近的 k 个邻居；2）通过最近邻的类别来确定文档 d_j 的类别。

延迟算法的一大优点在于其可以关注待分类文档 d_j 的特定特征，而不是像决策树那样的全局分类模型，后者可能会包含许多对 d_j 分类并不重要的特征。

1. **基本技术**

图 8-9 显示了一个 4 近邻分类器的例子（$k=4$）。其中有 4 个训练文档属于类别 c_a，而 6 个训练文档属于类别 c_b。待分类的文档 d_j 画成了黑色。在虚线圆之内的是与文档d_j 最近的 4 篇文档（距离是通过近似的度量函数计算出来的，如非归一化向量空间排序，参考 3.2.6 节）。在最近的 4 篇文档当中，三篇属于类别 c_b 而一篇属于类别 c_a，于是我们将文档 d_j 分配给类别 c_b。

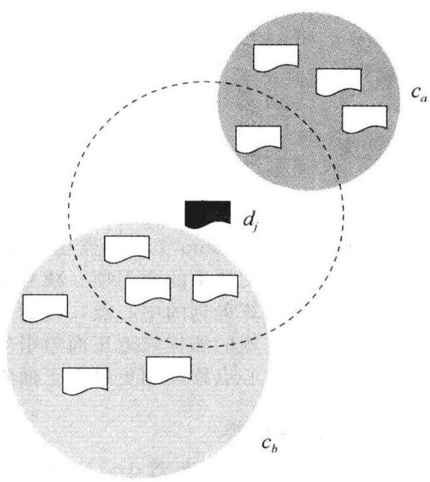


图 8-9 一个 4 近邻分类过程的例子

2. **文档分类**

在 k 近邻算法中，我们给每个文档-类别对 $[d_j, c_p]$ 赋予一个分数 S_{d_j, c_p}，计算方法为：

$$S_{d_j}, c_p = \sum_{d_t \in N_k(d_j)} similarity(d_j, d_t) \times \mathcal{T}(d_t, c_p) \tag{8-1}$$

其中 $N_k(d_j)$ 是 d_j 在训练集中 k 个最近邻的集合，$\mathcal{T}(d_t, c_p)$ 为定义在训练集上的函数，当文档 d_t 属于类别 c_p 时，返回 1；否则，返回 0。相似度函数可以根据向量模型或者非归一化向量模型的余弦公式进行计算，具体可参考 3.2.6 节。

分类新文档

对每篇文档 d_j，分类函数 $\mathcal{F}(d_j, c_p)$ 将其分配到分数 S_{d_j, c_p} 最高的类别 c_p 中。 299

k 近邻的一个问题在于其性能。为了确定最近的文档，分类器需要计算待分类文档与所有训练文档之间的距离。尽管可以采用特定目标的索引方法 [1707]，避免与训练集中的所有文档都进行比较，但 k 近邻分类的性能问题仍然存在。另一个问题在于如何选择最佳的 k 值，即寻找能够最优化分类结果的 k 值。

8.4.3 Rocchio 分类器

在 5.3 节中讨论的 Rocchio 相关反馈过程能够基于用户反馈修改原始的用户查询。其主要目的在于产生一个能够更好地逼近用户兴趣的新查询。Hull[797] 已经把 Rocchio 公式用于文本分类，正如我们下面要讨论的方式。

1. **基本技术**

Rocchio 相关反馈过程基于经典的向量模型，因此每篇文档 d_j 都可以表示为索引项权重向量 $\vec{d}_j$，形式为：

$$\vec{d}_j = (w_{1,j}, w_{2,j}, \cdots, w_{t,j})$$

其中 $w_{i,j}$ 表示索引项 k_i 在文档 d_j 中的权重，t 是词典的大小（可以参考 3.2.3 节的详细介绍）。

Rocchio 在文本分类上的应用是基于将训练集解释为反馈信息。在这种情况下，对于属

300 于某个给定类别 c_p 的训练文档中的索引项，我们认为其提供了正反馈；对于不属于类别 c_p 的训练文档中的索引项，我们认为其提供了负反馈（见图 8-10）。所有的反馈信息，即所有训练数据提供的类别成员信息，能够概括为索引项空间的中心点。中心点计算出来之后，新的测试文档可以通过与中心点的距离进行分类。

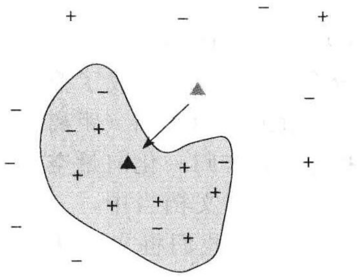


图 8-10 Rocchio 文本分类在文档索引项空间中的表示。加号表示属于某个给定类别 c_p 的训练文档中的索引项，减号表示不属于类别 c_p 的训练文档中的索引项。利用加权平均计算得到的中心点，我们用浅灰色的三角表示。如图所示，中心点可能并不如我们所期望的那样靠近正面索引项。通过修改正面索引项和负面索引项权重的方式，可以将中心点移动到更靠近正面索引项的位置。新的中心点位置用黑色三角表示

2. 文档分类

在 Rocchio 分类器中，类别 c_p 由权重向量 $\vec{c}_p=\{w_{1,p},w_{2,p},\cdots,w_{t,p}\}$ 表示，其中 t 为文档集中索引项的总数。设 n_p 为训练集中类别 c_p 的文档总数，N_t 为训练集的文档总数。于是，每个类别 c_p 的向量都可以通过计算其中心点得到。这里我们使用 Rocchio 公式（参考 5.3 节）的一个改进版本。

$$\vec{c}_p=\frac{\beta}{n_p}\sum_{d_j\in c_p}\vec{d}_j-\frac{\gamma}{N_t-n_p}\sum_{d_l\notin c_p}\vec{d}_l \tag{8-2}$$

需要注意的是，属于类别 c_p 的训练文档中的索引项得到了正的权重，不属于类别 c_p 的训练文档中的索引项得到了负的权重。

式（8-2）中的参数 β 和 γ 分别对在类别 c_p 中和不在 c_p 中的索引项的相对重要性进行建模，如图 8-10 所示。为了使中心点更靠近正面索引项而远离负面索引项，β 通常设置为比 γ 大得多的值。比如，在参考文献［290］中，$\beta=16$ 与 $\gamma=4$ 认为是标准取值，这一方案同时也被其他作者［406，837］所采用。另一种替代的方法是，设置 $\beta=1$ 与 $\gamma=0$，即认为只
301 由训练集中的正样本提供反馈，如文献［520，838，1444］所述。

分类新文档

为了对一个未知的新文档 d_j 进行分类，Rocchio 分类器根据文档和类中心的距离赋予每个文档-类别对 $[d_j,c_p]$ 分数 $S(d_j,c_p)$，具体形式为：

$$S(d_j,c_p)=|\vec{c}_p-\vec{d}_j|$$

在向量空间中，文档 d_j 越靠近类中心 $\vec{c}_p$，文档 d_j 属于类别 c_p 的可能性也就越高。因此，文档 d_j 会被分配到具有最高分数 $S(d_j,c_p)$ 的类别中去。

3. 基于查询区域的 Rocchio 分类器

在参考文献［1485］中，Singhal 等人研究了 Rocchio 公式负反馈的影响。他们提出，对于特定的领域，负反馈可能会使类别中心点远离用户感兴趣的主题，如图 8-11 所示。为了降低这种我们不希望看到的影响，他们提出应当减少用于负反馈的文档数量。也就是说，

他们认为应当只使用那些在所有作为负反馈的文档中“最正面”的文档，这样的文档我们称为近似正面文档（near-positive documents)。将负反馈限制在近似正面索引项上，可以确定一个能够更好地估计用户兴趣的查询区域。

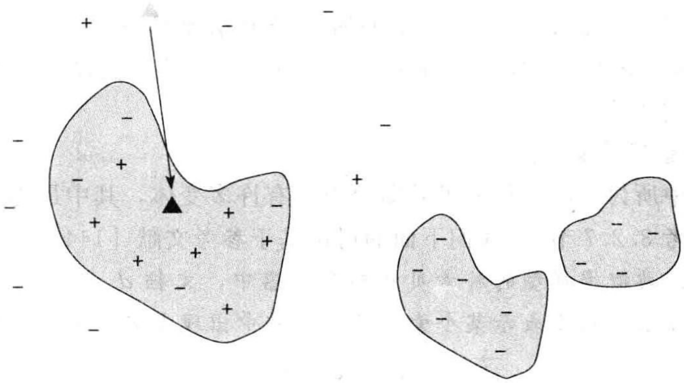


图 8-11 并非所有文档对描述一个给定类别 c_p 的特征都是有用的。即有些不属于类别 c_p 的文档应当不予考虑，因为它们会造成整个空间的失真。一种解决方法是只使用那些“最正面”的不属于类别 c_p 的文档

将“查询区域”这一想法用于文本分类，最早是由参考文献［1433］中提出的。作者认为应当按下面的方式选择近似正面文档。设 $\vec{c}_{p+}$ 为那些属于类别 c_p 的训练文档的中心点，即只考虑正面文档。对于所有不属于类别 c_p 的训练文档，即负面文档，统计它们与 $\vec{c}_{p+}$ 的距离。最后将那些与中心点距离较小的作为近似正面文档。基于这个思路的一些变体在参考文献［1201，1397，1676］中有介绍。 302

基于统计短语的查询区域方法，再加上其他的优化，可以使 Rocchio 分类器达到与现有的先进方法（如 boosting，参考 8.4.6 节）相同的性能，同时具有训练速度更快的优势［1433］。因此，基于查询区域的 Rocchio 分类器在文本分类领域是一个令人感兴趣且有竞争力的选择。

8.4.4 概率朴素贝叶斯文档分类

概率分类器给每个文档-类别对 $[d_j, c_p]$ 赋予一个表示该文档属于某个类别的概率。当我们计算了所有包含文档 d_j 的文档-类别对的概率时，分类器就把具有最高概率估计值的类别赋予 d_j。

1. **基本技术**

概率分类器与 3.2.7 节中讨论的经典概率模型类似。给定文档 d_j，它由二元权重 $w_{i,j}$ 组成的权重文档向量 $\vec{d}_j$ 表示，赋予每个文档-类别对 $[d_j, c_p]$ 一个概率 $P(c_p \mid \vec{d}_j)$，表示文档 d_j 属于类别 c_p 的概率。如果对于全部类别的概率都已经计算出来，就把文档分配到具有最高概率估计值的类别当中。

为了计算概率 $P(c_p \mid \vec{d}_j)$，概率分类器应用贝叶斯定理，具体形式如下：

$$P(c_p \mid \vec{d}_j) = \frac{P(c_p) \times P(\vec{d}_j \mid c_p)}{P(\vec{d}_j)} \tag{8-3}$$

其中 $P(\vec{d}_j)$ 为随机选择一篇文档而恰好返回由 $\vec{d}_j$ 表示的文档的概率，$P(c_p)$ 为随机选择一篇文档而该文档恰好属于类别 c_p 的概率。上面这些概率都作为归一化因子，而更重要的因子是概率 $P(\vec{d}_j|c_p)$，其计算需要能够高效地进行简化。最常见的简化方法是假设组成文档的所有索引项都是彼此独立的，正如所有经典信息检索模型中所假设的那样（参考第 3 章）。因为这一假设对真实文档并不成立，所以基于这一假设的分类器称为朴素贝叶斯（Naïve Bayes）分类器。

2. 二值独立朴素贝叶斯分类器

正如在 8.9 节中所讨论的，朴素贝叶斯分类器有许多变体，其中最为著名的基于经典概率模型的方法（参考 3.2.7 节）。我们下面的讨论基于参考文献 [1446]。

定义 在基于经典概率模型的朴素贝叶斯分类器中，文档 d_j 是由二值权重组成的向量

303 所表示的。每个维度上的权重表示某个索引项在文档中出现或者不出现。即有如下形式：

$$\vec{d}_j = (w_{1,j}, w_{2,j}, \cdots, w_{t,j})$$

其中 $w_{i,j}=1$，表示索引项 k_i（词）在文档 d_j 中出现；否则，$w_{i,j}=0$。对于每个文档-类别对 $[d_j, c_p]$，分类器赋予它一个分数 $S(d_j, c_p)$，形式为以下的比率：

$$S(d_j, c_p) = \frac{P(c_p|\vec{d}_j)}{P(\bar{c}_p|\vec{d}_j)}$$

其中 $P(c_p|\vec{d}_j)$ 是文档 d_j 属于类别 c_p 的概率，$P(\vec{c}_p|\vec{d}_j)$ 是文档 d_j 不属于类别 c_p 的概率。显然，$P(c_p|\vec{d}_j)+P(\vec{c}_p|\vec{d}_j)=1$。文档 d_j 最后被分配到具有最高分数 $S(d_j, c_p)$ 的类别中。

应用贝叶斯定理，我们得到

$$S(d_j, c_p) \sim \frac{P(\vec{d}_j|c_p)}{P(\vec{d}_j|\bar{c}_p)}$$

为了估计这些概率，我们采取了独立性假设，具体表述如下。

独立性假设 假设索引项 k_i（词）在文档 d_j 中的出现与文档中的其他索引项独立。在这个假设下，我们有

$$P(\vec{d}_j|c_p) = \prod_{k_i \in \vec{d}_j} P(k_i|c_p) \times \prod_{k_i \notin \vec{d}_j} P(\bar{k}_i|c_p)$$

和

$$P(\vec{d}_j|\bar{c}_p) = \prod_{k_i \in \vec{d}_j} P(k_i|\bar{c}_p) \times \prod_{k_i \notin \vec{d}_j} P(\bar{k}_i|\bar{c}_p)$$

其中 $P(k_i|c_p)$、$P(\vec{k}_i|c_p)$、$P(k_i|\vec{c}_p)$ 和 $P(\vec{k}_i|\vec{c}_p)$ 表示在属于（或不属于）类别 c_p 的文档中出现（或不出现）索引项 k_i 的概率。

我们可以应用和经典概率模型中的排序公式（参考 3.2.7 节）相同的推理方法进行推导，最终可以将分数 $S(d_j, c_p)$ 推导成以下形式。

$$S(d_j, c_p) \sim \sum_{k_i} w_{i,j}\left(\log\left(\frac{p_{iP}}{1-p_{iP}}\right)+\log\left(\frac{1-q_{iP}}{q_{iP}}\right)\right)$$

$$p_{iP} = P(k_i|c_p)$$

$$q_{iP} = P(k_i|\bar{c}_p)$$

其中 p_{iP} 表示第 i 个索引项是出自类别 c_p 中随机选择的文档的概率，q_{iP} 表示第 i 个索引项是出自类别 c_p 以外随机选择的文档的概率。 304

p_{iP} 和 q_{iP} 的概率可以通过训练文档集 $\mathcal{D}_t$ 估计得到，例如可用以下的估计方式，

$$p_{iP} = \frac{1 + \sum_{d_j | d_j \in \mathcal{D}_t \wedge k_i \in d_j} P(c_p | d_j)}{2 + \sum_{d_j \in \mathcal{D}_t} P(c_p | d_j)} = \frac{1 + n_{i,p}}{2 + n_p}$$

$$q_{iP} = \frac{1 + \sum_{d_j | d_j \in \mathcal{D}_t \wedge k_i \in d_j} P(\bar{c}_p | d_j)}{2 + \sum_{d_j \in \mathcal{D}_t} P(\bar{c}_p | d_j)} = \frac{1 + (n_i - n_{i,p})}{2 + (N_t - n_p)}$$

其中概率 $P(c_p | d_j) \in \{0,1\}$，$P(\vec{\bar{c}}_p | d_j) \in \{0,1\}$，可以从训练集中得到，变量 $n_{i,p}$、n_i、n_p 和 N_t 的定义参考 8.5.1 节。常数 1 和 2 用来避免当索引项没有出现在类别中时，可能出现的异常概率波动，正如文献 [1105] 中所建议的那样。

在 $S(d_j, c_p)$ 的计算过程中，只有那些在文档 d_j 中（即 $w_{i,j} > 0$）的索引项对这个分数有贡献。因为权重是二值的，而且将索引项的出现假设为是彼此独立的，所以这一分类器通常称为二值独立朴素贝叶斯分类器（binary independence Naïve Bayes classifier）。

分类新文档

在所有文档-类别对的分数 $S(d_j, c_p)$ 都计算完之后，分类器将每个文档分配到具有最高分数的类别中。

3. 多项朴素贝叶斯分类器

朴素贝叶斯分类器假设索引项的权重是二值的，即不考虑项频。考虑索引项的频率信息可以提高结果的质量，因此我们考虑使用这些信息来改进分类器。我们的讨论基于参考文献 [1105]。

为了对类别 c_p 中的文档 d_j 进行分类，我们使用式（8-3）。类别的先验概率形式为：

$$P(c_p) = \frac{\sum_{d_j \in \mathcal{D}_t} P(c_p | d_j)}{N_t} = \frac{n_p}{N_t} \tag{8-4}$$

其中 $P(c_p | d_j) \in \{0,1\}$，可以直接从大小为 N_t 的训练集中得到。文档的先验概率形式为：

$$P(\vec{d}_j) = \sum_{p=1}^{L} P_{prior}(\vec{d}_j | c_p) \times P(c_p) \tag{8-5}$$

其中 L 为类别总数，如前所示，而且

$$P_{prior}(\vec{d}_j | c_p) = \prod_{k_i \in \vec{d}_j} P(k_i | c_p) \times \prod_{k_i \notin \vec{d}_j} (1 - P(k_i | c_p))$$

$$P(k_i | c_p) = \frac{1 + \sum_{d_j | d_j \in \mathcal{D}_t \wedge k_i \in d_j} P(c_p | d_j)}{2 + \sum_{d_j \in \mathcal{D}_t} P(c_p | d_j)} = \frac{1 + n_{i,p}}{2 + n_p}$$ 305

正如之前对概率 p_{iP} 的计算。注意这些等式并没有考虑项频，而且只是用来估计先验概率。

为了估计最主要的概率 $P(\vec{d}_j | c_p)$，我们修改公式的形式使其包含项频，具体形式如下。

定义 假设类别 c_p 中的文档 d_j 是由一个随机过程生成的，在这个过程中索引项的抽取服从一个已知的分布。索引项的每次抽取我们都认为是一个伯努利试验，发生的概率为 $P(k_i|c_p)$。而且，每个索引项 k_i 的抽取次数就是其文档频率 $f_{i,j}$，即伯努利试验需要一直重复，直到所有索引项都出现在文档中。因此，文档中的索引项服从多项分布，即

$$P(\vec{d}_j|c_p) = F_j! \times \prod_{k_i \in d_j} \frac{[P(k_i|c_p)]^{f_{i,j}}}{f_{i,j}!} \tag{8-6}$$

其中 t 为词典大小，并且

$$F_j = \sum_{k_i \in d_j} f_{i,j}$$

注意 F_j 是文档 d_j 的索引项数量，即文档的长度。而索引项的概率可以从训练集中估计出来，如以下形式：

$$P(k_i|c_p) = \frac{\sum_{d_j \in \mathcal{D}_t} f_{i,j} P(c_p|d_j)}{\sum_{\forall k_i} \sum_{d_j \in \mathcal{D}_t} f_{i,j} P(c_p|d_j)} \tag{8-7}$$

其中 $\mathcal{D}_t$ 是训练文档集。而且，$P(c_p|d_j) \in \{0,1\}$ 可以直接从训练集中得到。

通过将式（8-4）～式（8-7）代入式（8-3），分类器就能够计算 $P(c_p|\vec{d}_j)$。

分类新文档

多项朴素贝叶斯分类器将每篇文档 d_j 根据式（8-3），分配到 $P(c_p|\vec{d}_j)$ 最大的类别中。

按照这样的方式，分类器考虑了项频信息，因此可能在一般文档集上带来更高的精度，正如参考文献［1105］中所介绍的那样。

8.4.5 支持向量机分类器

支持向量机（Support Vector Machine，SVM）分类器是一种相对较新颖的分类方法，由 Vapnik[429] 提出，并由 Joachims[838] 首先在文本分类中使用。由于支持向量机的复
306 杂性，在正式介绍其概念之前，我们首先阐述其技术背后的思想。

1. SVM 基本技术——直观认识

支持向量机（SVM）对两类分类问题提供了一种向量空间方法。所有的索引项形成了 t 维空间，而文档是其中的点（或者向量）。给定文档的向量表示后，任务就转化为找到一个决策面（即超平面），能够最好地区分 c_a 和 c_b 两个类别的元素。从训练数据中学习得到的超平面将空间分割成两个区域。所有类别 c_a 中的文档在其中一个区域，而类别 c_b 中的文档在另一个区域。在二维空间中，超平面是一条直线。在三维空间中，超平面是一个平面。在学习得到超平面后，新文档 d_j 可以通过计算与超平面的相对关系来进行分类。

举例来说，在一个简单的二维例子中，其训练数据点是线性可分的（即可以用一条直线进行分割），如图 8-12 所示。在所有能够将文档集正确地分成两个类别的直线中，直线 s 最大化了到每个类别中的文档的最短距离，因此我们认为这是最好的分割超平面。为了简便起见，我们称其为*决策超平面*（decision hyperplane）。注意直线 r 在这个例子中是一个较差的选择，因为它到类别 c_a 和类别 c_b 的最短距离较小。

在图 8-12 中，平行的虚线划定了寻找可行解的区域。为了简便起见，我们称其为划界

超平面集（delimiting hyperplanes)。恰好在划界超平面上的文档，我们称为支持向量(support vector)。穿过这个空间的直线都能够被选择为决策超平面的候选。平行于划界超平面的直线通常是最好的候选。在这个例子中，直线 s 将空间分割为两个相同的部分，被认为是最好的超平面，即决策超平面。

图 8-13 将图 8-12 中的例子显示在二维坐标系当中。类别 c_a 中的文档表示为方形点，类别 c_b 中的文档表示为圆形点。支持向量机优化问题可以按以下形式表述。 307

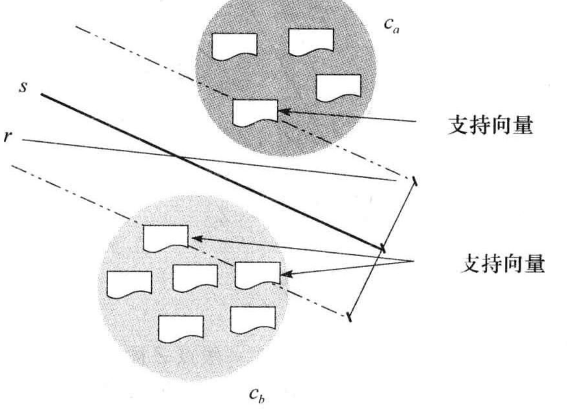


图 8-12 在这个二维的例子中，直线 s 最大化了到类别 c_a 和类别 c_b 中文档的最小距离（相对于直线 r)，并且构成了用于分类新文档的决策超平面

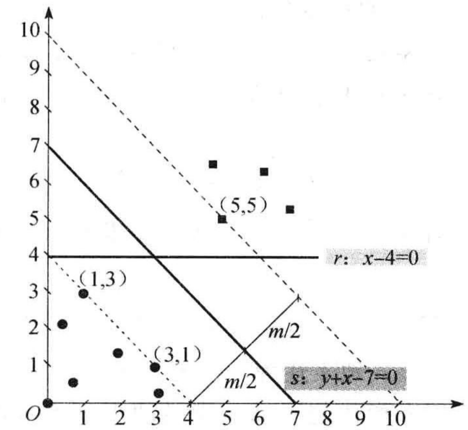


图 8-13 图 8-12 中显示的文档的图示。方形点表示类别 c_a 中的文档，圆形点表示类别 c_b 中的文档。支持向量为点（1，3)、(3，1)、(5，5)。直线 r 是一个合法的分割超平面，但直线 s 才是最佳的分割超平面——决策超平面

支持向量机优化问题 令 $\mathcal{H}_w$ 是能够将类别 c_a 中的文档与类别 c_b 中的文档分开的超平面。设 m_a 是在类别 c_a 中的文档与 $\mathcal{H}_w$ 的最近距离，m_b 是在类别 c_b 中的文档与 $\mathcal{H}_w$ 的最近距离，并有 $m_a + m_b = m$。距离 m 即是支持向量机的“分类间隔”(margin)。决策超平面 $\mathcal{H}_w$ 最大化了分类间隔 m。

在图 8-13 中，超平面 $r: x-4=0$ 将两个集合中的文档分割开来，与两个类别的最近文档是（1，3）和（5，5)，距离都是 1，因此其分类间隔为 2。而超平面 $s: y+x-7=0$ 对应的分类间隔是 $3\sqrt{2}$，在这个例子中是最大的，因此它就是决策超平面。在决策超平面确定后，新文档 d_j 可以通过检查它与决策超平面的相对位置来决定应该分类到 c_a 还是 c_b。

支持向量机可以形式化地表示为一个优化问题。我们首先回顾一下直线和超平面在 n 维空间中的向量符号表示。

2. 在 $\mathcal{R}^n$ 中的直线与超平面

定义 令 $\mathcal{R}^n$ 为以 O 为原点的 n 维空间。在 $\mathcal{R}^n$ 中，一般点 Z 被表示为向量 $\vec{z}$，形式为：

$$\vec{z} = (z_1, z_2, \cdots, z_n)$$ 308

其中 $z_i (1 \leqslant i \leqslant n)$ 为实数变量。我们对定点采用相似的表示方法，如对于 A、B、H、P、Q，我们分别用向量 $\vec{a}$、$\vec{b}$、$\vec{h}$、$\vec{p}$、$\vec{q}$ 来表示。但这些在 n 维空间中的向量是由实数常数组成的，而不是实数变量。

图 8-14 显示了在 $\mathcal{R}^n$ 中的一般点 Z 和定点 A 和 B 。

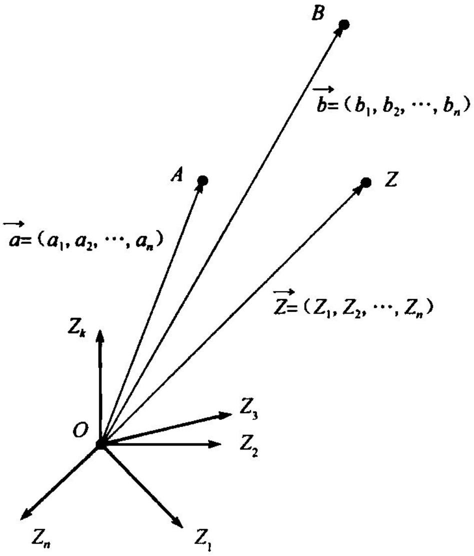


图 8-14 在 $\mathcal{R}^n$ 中，一般点 Z 被表示为向量 $\vec{z}$ ，其分量是多个实数变量 z_i 。对于定点如 A 和 B ，是由向量 $\vec{a}$ 和 $\vec{b}$ 表示的，其分量都是实数常数

图 8-15a 显示的是直线 s ，其方向为向量 $\vec{w}$ ，且过定点 P（表示为 $\vec{p}$），从而直线的参数方程可以表示为：

$$s: \vec{z} = t\vec{w} + \vec{p}$$

其中 $-\infty < t < +\infty$ 。由于 t 从 $-\infty$ 变化到 $+\infty$ ，因此点 $\vec{z}$ 遍历整条直线。

图 8-15b 显示的是超平面 $\mathcal{H}_w$ ，它包含定点 H ，且垂直于给定的向量 $\vec{w}$ 。该超平面的标准方程为：

$$\mathcal{H}_w: (\vec{z} - \vec{h})\vec{w} = 0$$

因为向量 $\vec{z} - \vec{h}$ 与 $\vec{w}$ 垂直。方程可以重新表示为：

$$\mathcal{H}_w: \vec{z}\vec{w} + k = 0 \tag{8-8}$$

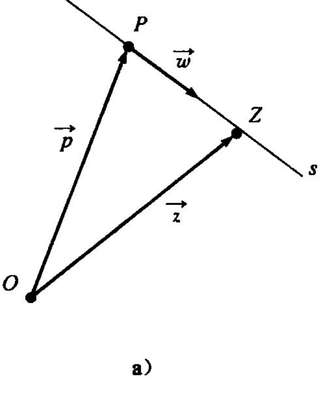
其中 $\vec{w}$ 与 $k = -\vec{h}\vec{w}$ 是待确定的。方程的第二种形式将 k 显式地表述为一维的变量，可以与标准方程
309 交换用来指代超平面。满足超平面方程的点 $\vec{z}$ 属于 $\mathcal{H}_w$ 。

图 8-16a 显示了从点 A 到超平面距离的线段 $\overline{AP}$ ，其中 P 是 A 在超平面上的投影。这个距离是按以下的方法计算的：由于点 A 和点 P 确定的直线是在 $\vec{w}$ 的方向上，其参数方程可以表示为：

$$line(\overline{AP}): \vec{z} = t\vec{w} + \vec{a}$$

其中 $-\infty < t < +\infty$ 。特别地，对于点 P ，我们有：

$$\vec{p} = t_p\vec{w} + \vec{a} \tag{8-9}$$



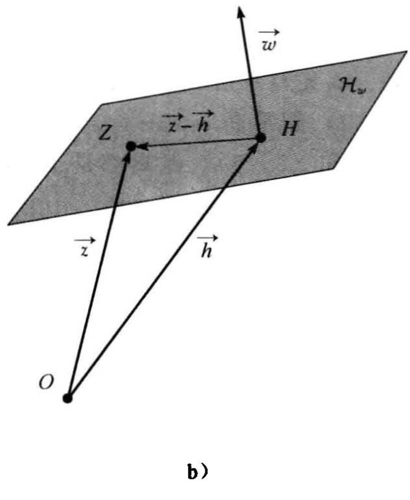


图 8-15 在 $\mathcal{R}^n$ 中的表示形式：a）直线 s ，其方向为向量 $\vec{w}$ ，且包含定点 $\vec{p} = (p_1, p_2, \cdots, p_n)$ ；b）超平面 $\mathcal{H}_w$ ，垂直于向量 $\vec{w}$ ，且包含定点 $\vec{h} = (h_1, h_2, \cdots, h_n)$ 。点 $\vec{z} = (z_1, z_2, \cdots, z_n)$ 表示 $\mathcal{R}^n$ 中的一般点（简便起见，原点 O 也是这么表示的）

其中 t_p 是点 P 所对应的 t 值。由于 $P \in \mathcal{H}_w$，可以将式（8-9）代入式（8-8），得到

$$(t_p\vec{w}+\vec{a})\vec{w}+k=0$$

解出 t_p

$$t_p=-\frac{\vec{a}\vec{w}+k}{|\vec{w}|^2}$$

其中 $|\vec{w}|$ 为其向量模长。通过将 t_p 代回式（8-9），有

$$\vec{a}-\vec{p}=\frac{\vec{a}\vec{w}+k}{|\vec{w}|}\times\frac{\vec{w}}{|\vec{w}|}$$ 310

由于 $\vec{w}/|\vec{w}|$ 是在 $\vec{w}$ 方向上的单位向量，则有

$$\overline{AP}=|\vec{a}-\vec{p}|=\frac{\vec{a}\vec{w}+k}{|\vec{w}|}$$

它是点 A 到超平面 $\mathcal{H}_w$ 的距离。

图 8-16b 显示了超平面方程的符号是如何根据超平面的位置改变的。超平面 $\mathcal{H}_w$ 上方的区域是由使得超平面方程 $\vec{z}\vec{w}+k$ 值为正的点 $\vec{z}$ 组成的，超平面 $\mathcal{H}_w$ 下方的区域是由使得超平面方程 $\vec{z}\vec{w}+k$ 值为负的点 $\vec{z}$ 组成的。

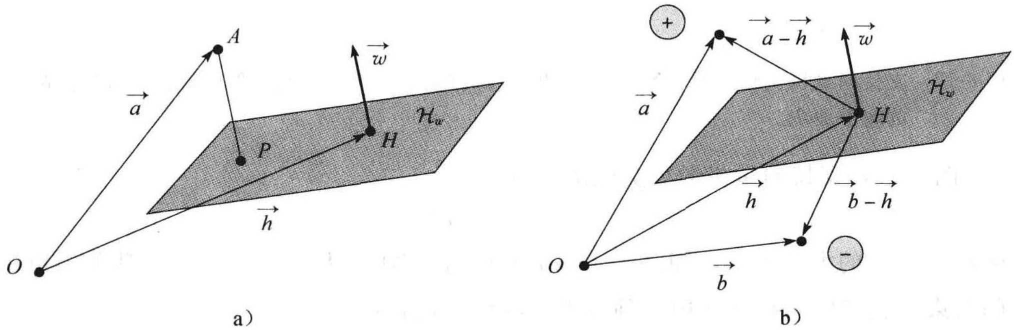


图 8-16 超平面 $\mathcal{H}_w$ 的性质：a）如果点 P 是点 A 在超平面 $\mathcal{H}_w$ 上的投影，那么线段 $\overline{AP}$ 就定义了点 A 到 $\mathcal{H}_w$ 的距离；b）在超平面上方的点使得方程为正值，在超平面下方的点使得方程为负值

3. SVM 基本技术——形式化表述

为了写出能够定义优化问题的方程组，我们首先观察图 8-17。坐标系的原点为点 O。点 A 和点 B 表示训练集中的两篇文档，分别属于划界超平面 $\mathcal{H}_a$ 和 $\mathcal{H}_b$，因此它们是支持向量。由于它们也是训练集的一部分，因此它们的类别是已知的，即我们知道点 A 关联到类别 c_a，而点 B 关联到类别 c_b。

分割超平面 $\mathcal{H}_w$ 在划界超平面界定的区域内，是由点 $\vec{h}$ 和法向量 $\vec{w}$ 共同确定的，这两个变量都是无法预先得到的。其方程形式为：

$$\mathcal{H}_w:\vec{z}\vec{w}+k=0$$

令 P 为点 A 在超平面 $\mathcal{H}_w$ 上的投影。点 A 到超平面的距离即是线段 $\overline{AP}$，即

$$\overline{AP}=\frac{\vec{a}\vec{w}+k}{|\vec{w}|}$$

现在让我们把注意力转移到超平面 $\mathcal{H}_b$ 上，它与 $\mathcal{H}_a$ 关于 $\mathcal{H}_w$ 对称。令 Q 为点 B 在超平面 $\mathcal{H}_w$ 上的投影。点 B 到超平面的距离即是线段 $\overline{BP}$，形式为： 311

$$\overline{BQ} = -\frac{\vec{b}\vec{w}+k}{|\vec{w}|}$$

其形式与线段 $\overline{AP}$ 类似。该式必定返回一个正数（这是一个距离度量），因为式 $\vec{b}\vec{w}+k$ 必定是负数，见图 8-16b。

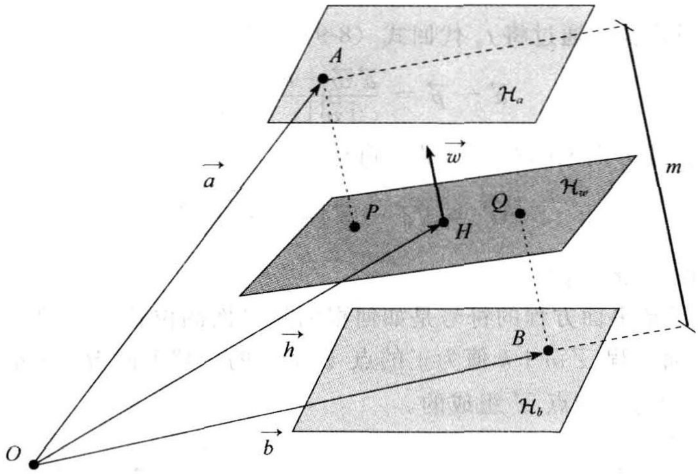


图 8-17 支持向量机优化问题：给定如 $\vec{a}$ 和 $\vec{b}$ 那样的支持向量，找到能够最大化分类间隔 m 的超平面 $\mathcal{H}_w$

根据图 8-17，支持向量机的分类间隔 m 为：

$$m = \overline{AP} + \overline{BQ}$$

表达式与 $\vec{w}$ 的大小无关，即许多不同大小的 $\vec{w}$ 都能最大化 m 。因此，我们能够引入对 $|\vec{w}|$ 的约束。支持向量机优化中一种常见的约束是设置：

$$\vec{a}\vec{w}+k=1$$

$$\vec{b}\vec{w}+k=-1$$

即我们限制解空间，要求向量 $\vec{w}$ 能使得超平面方程 $\vec{z}\vec{w}+k$ 对于所有支持向量都等于+1或者−1。需要注意的是，这也同样限制解向量所在的超平面恰好将分类间隔 m 等分。在这些条件下，分类间隔 m 的表达式为：

$$m = \frac{1}{|\vec{w}|} + \frac{1}{|\vec{w}|} = \frac{2}{|\vec{w}|}$$

对于一个与支持向量相比，与 $\mathcal{H}_w$ 距离更远的点 C 来说，必有 $\vec{z}\vec{w}+k>1$ 或者 $\vec{z}\vec{w}+$
312 $k<-1$ ，取决于该点位于正值区域还是负值区域。令 $\mathcal{T}=\{\cdots,[c_j,\vec{z}_j],[c_{j+1},\vec{z}_{j+1}],\cdots\}$ 为训练集，其中 c_j（c_a 或者 c_b）为关联到点 $\vec{z}_j$（即某个文档 d_j）的类别。类别只能取两个值，因为支持向量机是一个二值决策问题，于是有如下问题。

(1) 支持向量机优化问题

最大化 $m = 2/|\vec{w}|$

约束条件为

$$\vec{w}\vec{z}_j + k \geqslant +1 \quad c_j = c_a$$

$$\vec{w}\vec{z}_j + k \leqslant -1 \quad c_j = c_b$$

那些使得等式的值等于＋1 或者－1 的向量即是支持向量。

$|\vec{w}|$ 的计算需要涉及平方根（向量的模）。由于这个因素，该优化问题使用 $|\vec{w}|^2$ 会更简单。而且，这个问题还可以表述为一个最小化问题，如下所述。

（2）支持向量机优化问题

最小化 $m = \frac{1}{2}|\vec{w}|^2$

约束条件为

$$\vec{w}\vec{z}_j + k \geqslant +1 \quad c_j = c_a$$
$$\vec{w}\vec{z}_j + k \leqslant -1 \quad c_j = c_b$$

这与前面表述的形式有相同的解，但其优势在于，它能够表述为一个二次线性优化问题。这在线性规划中是一个著名问题，可以通过一种叫做二次规划的技术来高效地解决。对这种技术的讨论已经超出了本书的范畴，读者可以参考相关的材料。

例子　让我们再来考虑图 8-13 中的简单例子。在这个例子中，优化问题可以确定为以下形式（为简便起见，我们使用了最大化形式）：

最大化 $m = 2/|\vec{w}|$

约束条件为

$$\vec{w} \cdot (5,5) + k = +1$$
$$\vec{w} \cdot (1,3) + k = -1$$

在图 8-13 中的简单例子中，我们可以直接从几何角度来计算最大化的 m，并得到 $m = 3\sqrt{2}$（在实际应用中，m 的值是通过解上面的最大化问题得到的）。而且，如果将向量 $\vec{w}$ 表示为 (x,y)，那么有 $|\vec{w}| = \sqrt{x^2+y^2}$。因此有：

$$3\sqrt{2} = 2/\sqrt{x^2+y^2}$$
$$5x + 5y + k = +1$$
$$x + 3y + k = -1$$

从而有 $k = -5/3$ 或者 $k = -7/3$。对我们有用的值为 $k = -7/3$，从而有 $x = 1/3$，$y = 1/3$。因此，决策超平面的方程为以下形式： 313

$$(1/3,1/3) \cdot (x,y) + (-7/3) = 0$$

或

$$y + x - 7 = 0$$

即图 8-13 中直线 s 表示的方程。

4. 文档分类

对一篇新文档 d_j，表示为向量 $\vec{z}_j$，其分类过程是通过以下的决策函数完成的：

$$f(\vec{z}_j) = sign(\vec{w}\vec{z}_j + k)$$

如果 $f(\vec{z}_j)$ 的符号是正的，那么文档 d_j 分配到类别 c_a；否则，分配到类别 c_b。支持向量机分类器可能会有分类间隔的要求，以减少分类错误。在这种情况下，一篇新文档 d_j 被分类到类别 c_a 中，当且仅当 $\vec{w}\vec{z}_j + k > 1$；分类到类别 c_b 中，当且仅当 $\vec{w}\vec{z}_j + k < -1$。

当文档表示为多维权重向量时，支持向量机可以直接应用于文本分类。特别地，Joachims[838] 提出，支持向量机特别适合文档分类问题，因为它可以很好地处理高维而又十分稀疏的概念空间，且不容易发生过拟合。此外，他还表示应用特征选择来减少索引项数量不利于支持向量机的分类性能，因为事实上大多数特征都是与分类任务相关的。

5. 多类别的支持向量机

支持向量机只能进行二值决策，即判断一篇文档属于或者不属于某个给定的文档类。对于文本分类中的一般情形，也就是多类的情况，每一个类别都需要学习一个不同的分类器，我们将在下面进行讨论。

为了处理多类别的支持向量机分类问题，我们使用将多类问题分解为多个二类分类问题的策略。一种很自然的方式是对每个类别都建立一个二类分类问题。对于给定的类别 c_p，相应的二类分类器会尝试回答这样的问题："给定测试文档 d_j，其正确的类别标签是 c_p，还是其他 $(k-1)$ 个类别中的一个?"针对类别 c_p 的分类器是按以下方法训练的：将训练文档中所有属于类别 c_p 的作为正例样本，剩余的训练文档作为负例样本。这一策略也称为"一对多"(one-against-all)。

为了对一个新文档 d_j 进行分类，我们对每个类别都运行一次分类过程。如果有 L 个类别，那么分类过程就会重复 L 次。在每个过程中，不同的类别 c_p 与其他所有剩余的类别构成一个对于文档 d_j 的二值决策过程。这些过程都完成之后，文档 d_j 可能没有被分配给任何类别，或者分配给多个类别，甚至分配给所有的类别。为了决定究竟将文档 d_j 分配到哪个类别中，我们可以选择在 $\vec{d}_j$ 分类过程中使得分类间隔最大的那个类别。

另一种可能的方法是对任何一对类别 c_p 和 c_q 都构造一个分类器，把一个类别中的所有训练文档作为正例样本，而另一个类别中的所有样本都作为负例样本，剩余类别中的文档在这个分类器中被忽略。在这种情况下，需要回答的问题是："对于一篇给定的文档 $\vec{d}_j$，类
314 别 c_p 和 c_q 中哪个是正确的标签?"由于每一对类别都生成一个不同的分类器，这样一共会产生 $\frac{L(L-1)}{2}$ 个分类器，其中 L 是类别总数。为了对一个新实例进行分类，所有分类器的决策需要进行合并。简单的多数投票或者更复杂的合并策略都是可用的。这种策略通常也叫做"一对一"(one-against-one) 或者"全序对"(all-pairs) 方法。由于分类器的数量会随着类别数量的增加而呈平方级增长，因此这种方法在类别数量很大的情况下并不实用。

还存在其他可能的方案。比如可以使用一种投票策略对每篇文档都选择一个类别，具体请参考文献 [781]。

6. 线性不可分情况下的支持向量机方法

正如我们已经讨论过的那样，计算 $\vec{w}$ 的模意味着支持向量机是一个线性约束的二次优化问题。然而，这个问题在没有超平面能够将数据点分割成两个不相交集合的情况下是无解的，这种情况我们称之为*线性不可分* (non-linear separable)。在这种情况下，可以通过允许分类器犯一些错误（软分类间隔方法）或者将原始数据映射到一个高维空间，使得映射后数据线性可分的方法（核方法）来进行求解，我们将在下面的内容中具体讨论。

(1) 软分类间隔

软分类间隔方法是通过引入松弛变量 e_j 来实现的，这一变量用来衡量每个错分点（某个野值或者噪声样本，位于分类间隔之中或者在错误的分类区域中）与决策面的距离。这样就能将错分点引入的错误与付出的代价建立一个比例关系。这不仅有助于处理线性不可分的情形，还使得我们可以寻找能够正确分割大部分数据的分类面，同时不受野值或者噪声的影响。引入松弛变量后，支持向量机优化问题可以表述为：

最大化 $m = \frac{2}{|\vec{w}|} + \gamma \sum_j e_j$

约束条件为

$$\vec{w}\vec{z}_j + k \geqslant +1 - e_j \quad c_j = c_a$$
$$\vec{w}\vec{z}_j + k \leqslant -1 + e_j \quad c_j = c_b$$
$$\forall j \quad e_j \geqslant 0$$

为简便起见，这里我们又一次使用了最大化的形式。这一优化问题现在是在两个因素当中取得一个权衡：分类间隔大小与错分点带来的错误量。参数 γ 用来控制这两个因素的相对重要性，γ 的值越大，对于错分情形的容忍度就越低。

(2) *核方法*

用于处理线性不可分情形的核方法是通过在变换特征空间中寻找最大分类间隔超平面实现的，而映射后的数据是线性可分的。在变换空间中进行的线性操作，便能够有效地分割数据点，这就等价于在原始空间中进行非线性操作。 315

更形式化地说，支持向量机优化问题可以推广为以下形式：

支持向量机优化问题：

最小化 $m = \frac{1}{2} \times |\vec{w}|^2$

约束条件为

$$f(\vec{w}, \vec{z}_j) + k \geqslant +1 \quad c_j = c_a$$
$$f(\vec{w}, \vec{z}_j) + k \leqslant -1 \quad c_j = c_b$$

在实际应用中，空间变换可以通过使用不同的 f（核）函数来实现，而无须直接将数据点从原始空间映射到变换空间中。从这个意义上说，核函数可以视为一种关于输入向量 $\vec{w}$ 和 $\vec{z}_j$ 的相似度量。在传统的线性支持向量机的情况下，核函数 f 是输入向量的点积，即

$$f(\vec{w}, \vec{z}_j) = \vec{w}\vec{z}_j$$

然而，在一般情况下，许多其他函数都可以使用。常用的非线性核函数有：

- **多项式核函数**：$f(\vec{w}, \vec{x}_j) = (\vec{w}\vec{x}_j + 1)^d$，其中 d 为多项式的次数。
- **径向基函数**：$f(\vec{w}, \vec{x}_j) = \exp(\lambda \times |\vec{w}\vec{x}_j|^2)$，$\lambda > 0$。
- **sigmoid 函数**：$f(\vec{w}, \vec{x}_j) = \tanh(\rho(\vec{w}\vec{x}_j) + c)$，其中 $\rho > 0$，$c < 0$。

可以用做核函数的相似度量是有一定约束的，然而这已经超出了我们讨论的范围。

8.4.6　集成分类器

集成分类器将多个独立分类器的结果合并起来，希望能够提供高质量的结果。其背后的直观思想是基于独立的分类器都是足够准确而互异的。这里*准确*（accuracy）指的是分类器能够产生高于随机猜测水平的结果，*互异*（diversity）指的是不同的分类器对许多实例能产生不同的结果。

1. 基本技术

集成分类器将不同分类器的预测结果合并，产生一个新的预测分数。理想的情况是，合并后的分数能够带来比单个分类器更高精度的结果。直观的想法是，不同的分类器可能会有不同的表现，如果这些分类器都是准确而互异的话，那么将它们的决策恰当地合并可能会产生一个更有效的分类器。集成分类器在许多情况下都体现了实质性的优势，因此被视为机器学习领域的一项巨大进展［1429，1585］。 316

多年来，人们提出了许多集成学习的方法。这些方法的主要区别在于分类器的选择以及

合并预测的方式。合并预测的策略从简单的多数投票到基于独立分类器可靠性的加权平均，种类非常丰富。我们主要讨论其中的两种：叠加（stacking）与增强（boosting）。

2. **叠加型集成分类器**

一种提高由不同分类器产生的结果的方法是学习一个能够将单个分类器的预测合并集成的函数，正如参考文献［1716］和图 8-18 所显示的那样。在这种情况下，训练集中的每个文档-类别对 $[d_j, c_p]$ 都对应一个由不同分类器对该文档-类别对产生的预测结果所组成的向量。即我们使用向量式的二值决策，而不是单个的二值决策。这样，如果某个分类器做出错误的预测，也能够被其他分类器的预测纠正过来。元分类器的方法称为*叠加*（stacking）。其主要想法是将不同的分类器（通常称为*基分类器*，base classifier）产生的信息作为训练数据的元特征，提供给元分类器。元分类器并不直接预测给定文档 d_j 的类别，而是 1）预测能
317 够最好地预测 d_j 的基分类器；或者 2）将基分类器的预测合并，产生更好的结果。合并不同分类器预测的明显优势在于，某个基分类器的错误能够被其他基分类器所平衡。

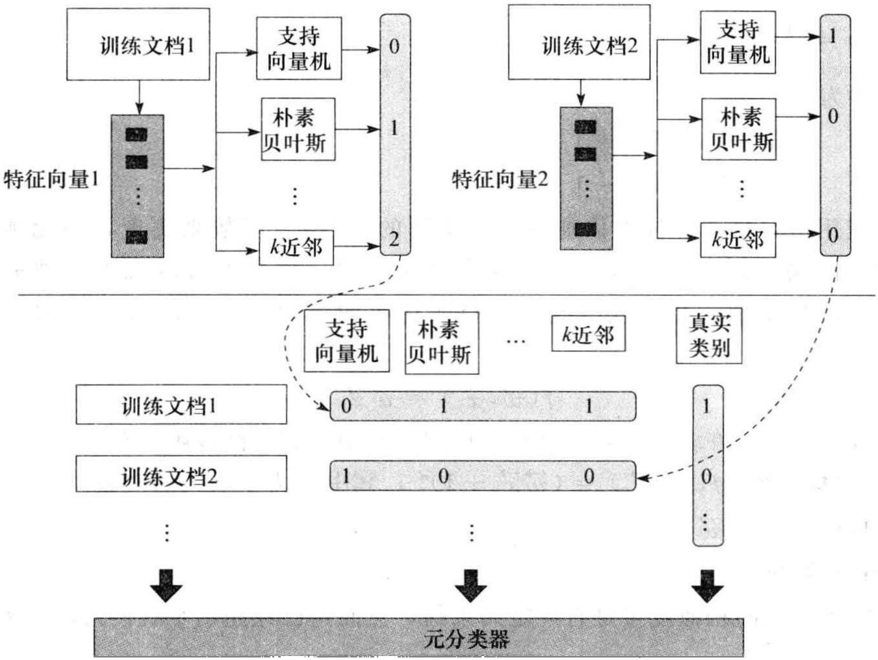


图 8-18 叠加型集成分类器的训练步骤。元分类器（meta classifier）是一个将所有分类器的预测合并的模型，希望能使模型更匹配训练集中的真实类别

分类新文档

一个在文本分类中应用叠加方法的例子显示在图 8-19。元分类器将不同分类器对于文档 d_x 和目标类别 c_p（图上未显示）的预测作为输入。然后将这些预测合并，产生自己的预测结果，这里是 $d_x \in c_p$。元分类器本身可能也是一个基分类器。比如支持向量机就常常是一种很好的元分类器的选择。

给定一篇新文档 d_j，每个基分类器都对其产生一个输出。将一个由所有预测结果组成的向量提供给元分类器以便产生最终的预测。如果想了解更多的细节，读者可以参考文献［256，1585，1716］。

3. **增强型集成分类器**

增强（boosting）是集成分类器采用的一种特殊方法，对于*相同的学习方法*，在多次迭

代后会产生多个分类器，该方法把这些分类器合并起来。在每次迭代中，使用叫做弱（weak）学习算法或者基（base）学习算法的方法来训练分类器，这些学习算法可能只产生有限的准确率（如单个决策规则或者决策树）。

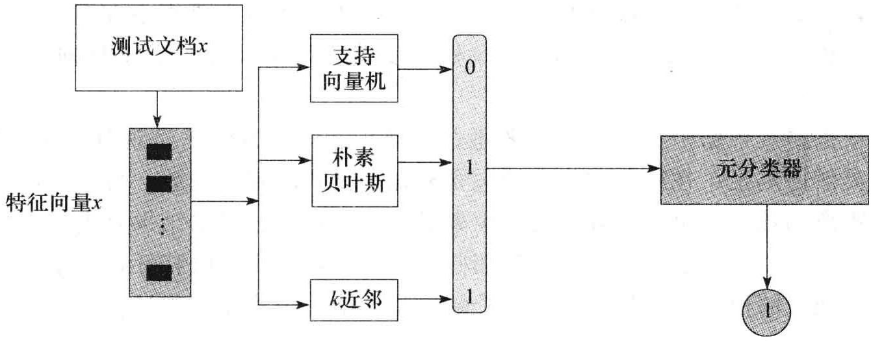


图 8-19 叠加型集成分类器的测试步骤。元分类器将不同分类器的预测作为输入，并将它们合并，产生关于文档 d_x 和目标类别 c_p（图上未显示）成员关系的元预测。在这个例子中，最终的预测是 1，即 $d_x \in c_p$

增强型集成分类器的主要想法是相继产生更好的分类器，而每个分类器都更重视在之前版本的基分类器中被错误分类的那些训练文档。在每次交互中，训练集中的每篇文档都赋予一个权重。利用这些权重选择在下次交互中需要重视的文档。为了保证训练集中的困难样本最终会变为“重要的”（focused）文档，没有正确分类的文档的权重在每轮迭代中都被加大。在 n 轮迭代后，多个训练好的分类器的输入通过加权平均来进行合并，其权重取决于每个分类器的估计错误率。 318

最初在文献［592］中提出的 AdaBoost 学习算法，是一个将增强学习（boosting）应用于文本分类的例子。在文献［1433］中讨论的一种算法变体显示在图 8-20 中。

AdaBoost

(1) let $\mathcal{T} : \mathcal{D}_t \times \mathcal{C}$ be the training set function
(2) let N_t be the training set size
(3) let M be the number of iterations
(4) initialize the weight w_j of each document d_j as $w_j = \frac{1}{N_t}$
(5) **for** $k \leftarrow 1 \ldots M$ **do**
(6) learn the classifier function $\mathcal{F}_k$ from the training set
(7) estimate weighted error:
$err_k = \sum_{d_j | d_j\, misclassified} w_j \;/\; \sum_{i=1}^{N_t} w_j$
(8) compute a classifier weight: $\alpha_k = \frac{1}{2} \times \log\left(\frac{1-err_k}{err_k}\right)$
(9) **for** all correctly classified examples e_j **do** $w_j \leftarrow w_j \times e^{-\alpha_k}$
(10) **for** all incorrectly classified examples e_j **do** $w_j \leftarrow w_j \times e^{\alpha_k}$
(11) normalize the weights w_j so that they sum up to 1

图 8-20 AdaBoost 的变体

分类新文档

为了对新文档 d_j 进行分类，我们将所有分类函数 $\mathcal{F}_k$ 都应用于该文档。文档 d_j 被分配到函数值最大的类别中。

细心的读者可能注意到，上面的形式只对二值分类问题有效。为了处理多类问题，需要将这一算法进行推广［1430］。比如一种叫做 AdaBoost. MH［1431］的推广方法，使用了 8.4.5 节中讨论的“一对多”策略，这意味着对于 K 个类别的问题就会有 K 个集成分类器。另一方面，

AdaBoost. M2[593] 使用了“一对一”的方法。其他方法可以参考文献 [33，786]。

8.4.7 关于监督算法的结束语

本节中，我们讨论了多种流行的监督文本分类算法。我们主要集中在图 8-2 中显示的六种监督算法，即决策树、最近邻、Rocchio 相关反馈、朴素贝叶斯、支持向量机以及集成学习。

决策树根据输入数据中存在的模式来建立模型，这样的方法是直观且易于使用。然而，这样的模型灵活性不足，在待分类的文档包含未知的模式时就会失效。另一方面，最近邻分类器是通过距离函数来找出与待分类文档距离最近的邻居。分类的结果是由最近邻的类别决
319 定的。这种方法的一大优点是可以只关注那些待分类文档中出现的特征，而不需要查找所有输入数据中出现过的模式。这种方法的一个缺点是性能，因为需要现场计算与其他许多文档的距离。

Rocchio 分类器利用相关反馈的理论来计算类中心。通过修改索引项权重，这一方法能够在文档索引项空间中进行重定位，从而更好地反映正面索引项的影响。正如在基于查询区域的 Rocchio 分类器中所实现的，需要更多地考虑那些属于正面文档的索引项。这种优化策略能够对 Rocchio 分类器进行精细的调整，从而得到高质量的结果。高级 Rocchio 分类器可以达到最好的性能，具有很强的竞争力。

朴素贝叶斯分类器利用概率论来估计正确分类的可能性。这种方法通常会有不错的表现，能够作为一种有竞争力的文本分类方法。支持向量机分类器天然地构成了一个优化问题的形式。这一形式考虑了分类错误间隔，能够有效地避免分类错误。这种方法通常能够训练出高质量的分类器，单独的方法（非集成的）一般很难有其他方法可以匹敌。

集成分类器将不同类型的分类器合并，产生高质量的结果。这种方法在训练和测试时都是十分昂贵的。然而，它通常能带来最好的结果，因此也常常作为任务的基线，尤其在希望能够对方法做全面研究的时候。

8.5 特征选择或降维

正如在 8.4 节中讨论的，庞大的特征空间可能导致文档分类器无法实际应用，因为对新文档的分类可能需要消耗太多的时间。这个问题的传统解决方法是选择所有特征的一个子集来表示文档，从而减小特征空间的维度。这个步骤称为*特征选择*（feature selection），我们将在这一节进行讨论。

为了产生特征向量，我们需要选择特征（索引项）的一个子集来表示文档。除了降低文档表示的维度之外，特征选择也有助于避免*过拟合*（overfitting）的情况，即降低了模型过于特例化的风险。过分的特例化可能会导致模型无法很好地推广到未知的新文档。我们的讨论基于参考文献 [1744]。

假设在索引过程中，每个索引项-文本对 $[k_i, d_j]$ 都与一个权重 $w_{i,j}$ 相关联。这个权重可以简单地设置为索引项的文档频率，即包含索引项 k_i 的不同文档的数量。或者也可以设置为 TF-IDF 权重（更多细节可以参考 3.2.4 节）。给定权重 $w_{i,j}$ 后，我们从中选择最终组成特征向量的索引项。这一过程通常称为*基于索引项选择的降维*（dimensionality reduction by term selection），可以通过不同的标准来实现。这里我们主要讨论五种标准：索引项文档频率、TF-IDF 权重、互信息、信息增益以及卡方检验。

在正式开始讨论之前，我们首先给出项-类别出现列联表的定义，它与 3.2.7 节中经典概率模型的列联表是类似的。 320

8.5.1 项-类别出现列联表

如图 8-5 所示，所有的索引项组成了文档表示。为了降低文档表示的大小，需要进行特征选择过程。这一过程在很大程度上依赖于对索引项在文档和类别中出现次数的统计。具体表述如下。

定义 令 $\mathcal{D}_t$ 为训练文档组成的集合，N_t 为 $\mathcal{D}_t$ 中的文档数量，n_i 为 $\mathcal{D}_t$ 包含索引项 k_i 的文档数量（值得注意的是，n_i 在第 3 章指的是在整个文档集中包含索引项 k_i 的文档，而这里指的是训练集中包含索引项 k_i 的文档）。另外，令 $C=\{c_1,c_2,\cdots,c_L\}$ 为总共 L 个类别的集合。假设存在一个训练集函数 $\mathcal{T}:\mathcal{D}_t\times C\rightarrow[0,1]$，令 n_p 为属于类别 $c_p\in C$ 的文档的数量。项-类别出现列联表即为以下形式：

情形	在类别 c_p 中的文档	不在类别 c_p 中的文档	共计
包含 k_i 的文档	$n_{i,p}$	$n_i-n_{i,p}$	n_i
不包含 k_i 的文档	$n_p-n_{i,p}$	$N_t-n_i-(n_p-n_{i,p})$	N_t-n_i
所有文档	n_p	N_t-n_p	N_t

包含索引项 k_i 且属于类别 c_p 的文档数量记为 $n_{i,p}$，包含索引项 k_i 且不在类别 c_p 中的文档数量为 $n_i-n_{i,p}$。n_p 为属于类别 c_p 的训练文档的总数，$n_p-n_{i,p}$ 为类别 c_p 中不包含索引项 k_i 的文档数量。其余的数量也都以类似的方式计算。项-类别出现列联表显式地展现了组成文档训练集的各个子集的大小。

有了上面的项-类别出现列联表之后，我们就可以定义我们感兴趣的各种概率，具体形式如下。

令 d_j 为随机选自训练集的一篇文档。定义：

$P(k_i)$：$k_i\in d_j$ 的概率。

$P(\overline{k}_i)$：$k_i\notin d_j$ 的概率。

$P(c_p)$：$d_j\in c_p$ 的概率。

$P(\overline{c}_p)$：$d_j\notin c_p$ 的概率。

$P(k_i,c_p)$：$k_i\in d_j$ 且 $d_j\in c_p$ 的概率。

$P(\overline{k}_i,c_p)$：$k_i\notin d_j$ 且 $d_j\in c_p$ 的概率。

$P(k_i,\overline{c}_p)$：$k_i\in d_j$ 且 $d_j\notin c_p$ 的概率。

$P\overline{k}_i,\overline{c}_p)$：$k_i\notin d_j$ 且 $d_j\notin c_p$ 的概率。 321

这些概率可以按如下形式计算：

$$P(k_i)=\frac{n_i}{N_t}\qquad P(\overline{k}_i)=\frac{N_t-n_i}{N_t}$$

$$P(c_p)=\frac{n_p}{N_t}\qquad P(\overline{c}_p)=\frac{N_t-N_p}{N_t}$$

$$P(k_i,c_p)=\frac{n_{i,p}}{N_t}\qquad P(\overline{k}_i,c_p)=\frac{n_p-n_{i,p}}{N_t}$$

$$P(k_i,\overline{c}_p)=\frac{n_i-n_{i,p}}{N_t}\qquad P(\overline{k}_i,\overline{c}_p)=\frac{N_t-n_i-(n_p-n_{i,p})}{N_t}$$

这些概率都很重要，它们在本节当中讨论的各种索引项选择方法中都会被使用。

8.5.2 索引项文档频率

一种简单而有效的索引项选择过程只使用那些文档频率超过某个预定义的频率阈值的索引项来表示文档，即

基于索引项文档频率的特征选择。设 K_{th} 为索引项文档频率的阈值。那么，所有满足 $n_i \geq K_{th}$ 的索引项 k_i 被保留下来，其他索引项都被舍弃。文档表示根据那些保留的索引项来重新计算。

本节所讨论的特征选择过程都应当在去除了禁用词之后再进行。即使是简单的高频文档索引项选择的方法也能有效地降低空间的维度，同时不会对最终的效果有太大的影响。

8.5.3 TF-IDF 权重

一种类似但更为精细的过程是保留那些在每篇文档 d_j 中 TF-IDF 权重较高的索引项（参考第 3 章关于 TF-IDF 权重更为详细的讨论）。

基于 TF-IDF 权重的特征选择。设 $w_{i,j}$ 为项-文档对 $[k_i, d_j]$ 的 TF-IDF 权重，与 3.2.4 节的定义类似。另外，令 K_{th} 为 TF-IDF 权重的阈值。从而所有满足 $n_i \geq K_{th}$ 的索引项 k_i 被保留下来，其他索引项都被舍弃。文档表示根据那些保留的索引项来重新计算。

这一过程是对上一个过程的补充。频率很低的索引项，其 TF-IDF 权重也很低，在这两个过程中都会被舍弃。另一方面，对于高频的索引项就会有不同的处理方式。在基于索引项文档频率的选择方法中会保留它们，但基于 TF-IDF 权重的选择方法可能会将它们去除，因为高频可能会导致较低的 TF-IDF 权重。在文献 [1744] 中的实验表明，使用 TF-IDF 权重进行特征选择能够将空间维度减小 10 倍而不带来任何效果上的损失。甚至可以在将空间维
322 度缩小 100 倍的情况下，只对效果产生很小的影响。

8.5.4 互信息

互信息是两个随机变量分布相对熵的度量。如果这些变量之间是独立的，那么它们的互信息为 0，即关于一个变量的知识不能推理出另一个变量的任何信息。索引项 k_i 与类别集合 C 的互信息 $MI(k_i, C)$ 可以表示为如下期望值：

$$I(k_i, c_p) = \log\left(\frac{P(k_i, c_p)}{P(k_i)P(c_p)}\right) = \log\left[\frac{\frac{n_{i,p}}{N_t}}{\frac{n_i}{N_t} \times \frac{n_p}{N_t}}\right]$$

再对所有类别求期望值，即有，

$$MI(k_i, C) = \sum_{p=1}^{L} P(c_p) I(k_i, c_p) \tag{8-10}$$

$$= \sum_{p=1}^{L} \frac{n_p}{N_t} \log\left[\frac{\frac{n_{i,p}}{N_t}}{\frac{n_i}{N_t} \times \frac{n_p}{N_t}}\right]$$

另一种方法是使用所有类别中索引项互信息最大的那个，即

$$I_{max}(k_i, C) = \max_{p=1}^{L} P(k_i, c_p)$$

$$= \max_{p=1}^{L} \log \left(\frac{\frac{n_{i,p}}{N_t}}{\frac{n_i}{N_t} \times \frac{n_p}{N_t}} \right)$$

在这种情况下，特征选择过程可以概述为：

基于熵的特征选择。设 K_{th} 为熵的阈值。那么所有满足 $MI(k_i, C) \geqslant K_{th}$（或者 $I_{max}(k_i, C) \geqslant K_{th}$）的索引项 k_i 被保留下来，其他索引项都被舍弃。文档表示根据那些保留的索引项来重新计算。

8.5.5 信息增益

信息增益这一测度是对互信息的补充。它不仅考虑索引项在文档中出现的概率，同时也考虑索引项不在文档中出现的概率。因此它平衡了索引项与文档共现和索引项不与文档共现两种情况对其信息量的影响。

索引项 k_i 对于类别集合 C 的信息增益 $IG(k_i, C)$ 定义如下：

$$IG(k_i, C) = H(C) - H(C|k_i) - H(C|\neg k_i)$$

其中 $H(C)$ 为类别集合 C 的熵，$H(C|k_i)$ 与 $H(C|\neg k_i)$ 为索引项 k_i 出现或者不出现情况下 C 的条件熵。从信息论角度上来说，$IG(k_i, C)$ 用来度量在知道 k_i 的信息之后，所增加的关于 C 的知识。 323

使用在 8.5.1 节中定义的概率以及熵的表达式（见 6.5.1 节），有：

$$\begin{aligned} IG(k_i, C) = & -\sum_{p=1}^{L} P(c_p) \log P(c_p) \\ & -\left(-\sum_{p=1}^{L} P(k_i, c_p) \log P(c_p | k_i)\right) \\ & -\left(-\sum_{p=1}^{L} P(\bar{k}_i, c_p) \log P(c_p | \bar{k}_i)\right) \end{aligned}$$

应用贝叶斯法则，可以将其重写为以下形式：

$$\begin{aligned} IG(k_i, C) = & -\sum_{p=1}^{L} \left(P(c_p) \log P(c_p) - P(k_i, c_p) \log\left(\frac{P(k_i, c_p)}{P(k_i)} \right) \right. \\ & \left. - P(\bar{k}_i, c_p) \log\left(\frac{P(\bar{k}_i, c_p)}{P(\bar{k}_i)} \right) \right) \end{aligned}$$

通过使用在 8.5.4 节中的概率定义，可以将其改写为：

$$IG(k_i, C) = -\sum_{p=1}^{L} \left(\frac{n_p}{N_t} \log\left(\frac{n_p}{N_t} \right) - \frac{n_{i,p}}{N_t} \log\left(\frac{n_{i,p}}{n_i} \right) - \frac{n_p - n_{i,p}}{N_t} \log\left(\frac{n_p - n_{i,p}}{N_t - n_i} \right) \right) \quad (8\text{-}11)$$

这一特征选择的过程可以概述如下。

基于信息增益的特征选择。设 K_{th} 为信息增益的阈值。那么，所有满足 $IG(k_i, C) \geqslant K_{th}$ 的索引项 k_i 被保留下来，其他索引项都被舍弃。文档表示根据那些保留的索引项来重新计算。

8.5.6 卡方检验

卡方检验是对索引项 k_i 和类别 c_p 独立性的缺失所做的度量。这一统计量定义如下：

$$\chi^2(k_i, c_p) = \frac{N_t (P(k_i, c_p) P(\neg k_i, \neg c_p) - P(k_i, \neg c_p) P(\neg k_i, c_p))^2}{P(k_i) P(\neg k_i) P(c_p) P(\neg c_p)}$$

324 通过使用在8.5.1节中的概率定义，可以将其改写为：

$$\chi^2(k_i,c_p)=\frac{N_t(n_{i,p}(N_t-n_i-n_p+n_{i,p})-(n_i-n_{i,p})(n_p-n_{i,p}))^2}{n_p(N_t-n_p)n_i(N_t-n_i)}$$
$$=\frac{N_t(N_tn_{i,p}-n_pn_i)^2}{n_pn_i(N_t-n_p)(N_t-n_i)}$$

为了对索引项 k_i 进行特征选择，我们需要按以下形式计算索引项的平均（或者最大）卡方检验值。

$$\chi^2_{avg}(k_i)=\sum_{p=1}^{L}P(c_p)\chi^2(k_i,c_p) \qquad (8\text{-}12)$$
$$\chi^2_{max}(k_i)=\max_{p=1}^{L}\chi^2(k_i,c_p)$$

这一特征选择的过程可以概述如下。

基于卡方检验的特征选择。 令 K_{th} 为卡方检验的阈值。那么，所有满足 $\chi^2_{avg}(k_i)\geqslant K_{th}$（或者 $\chi^2_{max}(k_i)\geqslant K_{th}$）的索引项 k_i 被保留下来，其他索引项都被舍弃。文档表示根据那些保留的索引项来重新计算。

8.5.7 特征选择的作用

特征选择能够将索引项空间即特征数的大小减少一到两个数量级，而不会对文本分类的过程造成太大的影响。有些特征选择的方法会在平均水平上比其他方法表现得更好。如文献[1446]的实验就说明了卡方检验和信息增益的指标相对于其他三种指标的优越性。

特征选择所带来的维度约减是至关重要的。这样才能允许我们实现更为精细的文本分类算法，从而改进结果。而这些算法在原始索引项空间中是无法实现的，因为这需要很长的运行时间，使得其不能实际应用。

8.6 评价指标

与所有信息检索技术一样，评价在任何文本分类方法的发展中都是非常重要的一个步骤。没有合理的评价与基准，就没有办法判断一个新提出的文本分类器的好坏。评价是验证新提出的分类方法的一个重要步骤。本节中，我们将介绍一些评价单标签文本分类器时最常用的指标。我们的讨论受到参考文献[1446，1743]很大的影响。

8.6.1 列联表

为了描述文本分类中最常用的一些评价指标，我们首先定义对于某个类别 c_p 的列联表。它与8.5.1节中的项-类别出现列联表是类似的，具体定义如下。

定义 设 $\mathcal{D}$ 为文档集，N_t 为所有训练文档（或者所有已知类别的测试文档）组成的集合，N_t 为 $\mathcal{D}_t$ 中的文档数量。另外，设 $\mathcal{C}=\{c_1,c_2,\cdots,c_L\}$ 为总共 L 个类别的集合。假设存在一个训
325 练集函数 $\mathcal{T}:\mathcal{D}_t\times\mathcal{C}\rightarrow[0,1]$ 和一个文本分类函数 $\mathcal{F}:\mathcal{D}\times\mathcal{C}\rightarrow[0,1]$。设 n_t 为被函数 $\mathcal{T}$ 分配到类别 c_p 的文档的数量（我们早先设其为 n_p），n_f 为被函数 $\mathcal{F}$ 分配到类别 c_p 的文档的数量。假设对训练集中的所有文档都应用了该分类器。那么，我们可以构造以下形式的列联表：

情形	$\mathcal{T}(d_j,c_p)=1$	$\mathcal{T}(d_j,c_p)=0$	总计
$\mathcal{F}(d_j,c_p)=1$	$n_{f,t}$	$n_f-n_{f,t}$	n_f
$\mathcal{F}(d_j,c_p)=0$	$n_t-n_{f,t}$	$N_t-n_f-n_t+n_{f,t}$	N_t-n_f
所有文档	n_t	N_t-n_t	N_t

其中 $n_{f,t}$ 是在训练函数和分类函数中都分配到类别 c_p 中的文档数量。值得注意的是，我们改变了符号表示方法，使得分类函数和训练函数产生的不同集合得以区分。另外为了简化我们的符号表示，我们没有在下标中包含类别 c_p 的引用。给定上面的列联表之后，类别 c_p 中的训练文档被错分的数量为 $n_t - n_{f,t}$ 。剩余的数值都可以使用相似的方法计算出来。

8.6.2　准确率和错误率

文本分类器对于某个给定类别 c_p 的准确率和错误率指标是按如下方式定义的：准确率是训练文档中被分类器分配到正确类别的文档所占的比例；错误率是训练文档中被分类器分配到不正确类别的文档所占的比例。即

$$Acc(c_p) = \frac{n_{f,t} + (N_t - n_f - n_t + n_{f,t})}{N_t} \tag{8-13}$$

$$Err(c_p) = \frac{(n_f - n_{f,t}) + (n_t - n_{f,t})}{N_t} \tag{8-14}$$

注意到必然有：

$$Acc(c_p) + Err(c_p) = 1$$

准确率和错误率尽管使用很广泛，但仍然存在着很大的不足。假设有这样一个例子，一个二类分类问题只有两个类别 c_p 和 c_r 。如果某个类别的文档数量相对于文档集合的总数来说比例很小的，那么这些指标可能就会变得不可信。举例来说，假设在 1000 个文档中，只有 20 个属于类别 c_p 。那么，一个非常简单的分类器——仅仅简单的假设所有文档都不属于类别 c_p 就能获得 98%（980/1000）的准确率和 2%的错误率。这样的数值表示我们拥有了一个很好的分类器，但这与实际情况不符。

假设现在另一个分类器能正确地预测类别 c_p 中 50%的文档，如表 8-2 所示。在这种情况下，准确率和错误率为：

表 8-2　某个分类器的列联表，该分类器能够正确地预测类别 c_p 中 20 个文档的一半

情形	$\mathcal{T}(d_j,c_p)=1$	$\mathcal{T}(d_j,c_p)=0$	总计
$\mathcal{F}(d_j,c_p)=1$	10	0	10
$\mathcal{F}(d_j,c_p)=0$	10	980	990
所有文档	20	980	1000

$$Acc(c_p) = \frac{10+980}{1000} = 99\%$$

$$Err(c_p) = \frac{10+0}{1000} = 1\%$$ 326

这个分类器比只猜测所有文档都不属于类别 c_p 的分类器要好得多，但其准确率只提高了 1%（从 98%增加到 99%）。表示这两个分类器几乎是等价的，但这与实际情况不符。

8.6.3　精度和召回率

文本分类中的精度（precision）与召回率（recall）是信息检索中精度率与召回率指标的翻版。它们用来衡量文本分类器的质量，而且能把某些我们前面提到的关于准确率的问题降低到最小。

对于某个给定类别 c_p 的精度与召回率数值是按以下方法计算的。

$$P(c_p) = \frac{n_{f,t}}{n_f} \tag{8-15}$$

$$R(c_p) = \frac{n_{f,t}}{n_t} \tag{8-16}$$

精度是所有被分类器分配到类别 c_p 的文档中确实属于类别 c_p 的文档（根据测试集）所占的比例。召回率是所有确实属于类别 c_p 的文档（根据测试集）中被分类器正确分配到类别 c_p 的文档所占的比例。

再回到我们用来计算准确率的那个小集合的例子。假设训练集中只有 20 个文档属于类别 c_p 。简单地认为所有文档都不属于类别 c_p 的朴素分类器会导致精度为 0，而对于图 8-2 所示的分类器，精度和召回率数值为：

$$P(c_p) = \frac{10}{10} = 100\%$$

$$R(c_p) = \frac{10}{20} = 50\%$$

即分类器对于类别 c_p 有 100%的精度和 50%的召回率。

通常，把精度和召回率合并为一个指标会更便于使用。一种最常用的指标叫做 F 测度或 F 值，我们将在 8.6.4 节进行讨论。

8.6.4 F 测度和 F_1

F 测度 [1743] 将精度和召回率合并起来，而且可以对这些指标赋予不同的权重（参考 4.3.2 节）。具体形式如下。

327

$$F_\alpha(c_p) = \frac{(\alpha^2 + 1)P(c_p)R(c_p)}{\alpha^2 P(c_p) + R(c_p)} \tag{8-17}$$

其中 α 定义了精度和召回率的相对重要性。当 $\alpha = 0$ 时，只考虑精度；当 $\alpha = \infty$ 时，只考虑召回率；当 $\alpha = 0.5$ 时，召回率的重要性被认为是精度的一半；以此类推。

最常用的 F 测度形式是通过赋予精度和召回率相同的权重得到的，即令 $\alpha = 1$ 。这一指标也被称为 F_1 测度，计算形式如下：

$$F_1(c_p) = \frac{2P(c_p)R(c_p)}{P(c_p) + R(c_p)} \tag{8-18}$$

F_1 平衡了精度和召回率的相对重要性，对于表 8-1 所示的例子，有：

$$F_1(c_p) = \frac{2 \times 1 \times 0.5}{1 + 0.5} \sim 67\%$$

宏平均与微平均 F_1

通过计算所有类别的平均 F_1 值，我们也常常赋予分类器一个唯一的 F_1 值。在文献 [1743] 中介绍了两种平均函数：微平均 F_1（micro-average F_1，micF_1 ）和宏平均 F_1（macro-average F_1，macF_1 ）。接下来将进行具体讨论。

宏平均 F_1 计算方法为：

$$macF_1 = \frac{\sum_{p=1}^{|C|} F_1(c_p)}{|C|} \tag{8-19}$$

因此，宏平均 F_1 简单地对所有类别的 F_1 值取平均。

为了计算微平均 F_1 ，我们需要得到基于所有类别的精度和召回率，即

$$P = \frac{\sum_{c_p \in C} n_{f,t}}{\sum_{c_p \in C} n_f}$$

$$R=\frac{\sum_{c_p\in C}n_{f,t}}{\sum_{c_p\in C}n_t}$$

计算出全局精度和召回率之后，微平均 F_1 的计算形式为：

$$micF_1=\frac{2PR}{P+R} \tag{8-20}$$

在微平均 F_1 中，每一篇文档都被赋予相同的重要性。在宏平均 F_1 中，每一个类别都被赋予相同的重要性。结论是宏平均 F_1 能够更好地刻画分类器在多类别情形下的性能。当类别分布非常不均匀时，这一点是十分重要的。这种情况下，两种平均指标都应当考虑。 328

8.6.5 交叉检验

交叉检验（cross-validation）已经成为对分类结果进行统计验证的标准方法［1140，1446］。它构建 k 个不同的分类器：$\Psi_1,\Psi_2,\cdots,\Psi_k$。为了检验分类器，我们将文档训练集 $\mathcal{D}_t$ 分割为 k 个不相交的集合（或折，fold），每折的大小为：$N_{t1},N_{t2},\cdots,N_{tk}$。分类器 Ψ_i 将大小为 N_{ti} 的第 i 折作为测试集，$\mathcal{D}_t$ 中剩余的、大小为 N_t-N_{ti} 的文档作为训练集。

每个分类器都用相同的评价指标进行独立的评测，如精度、召回率或 F_1 值。最终通过计算 k 个指标的平均值来完成交叉检验。最常用的 k 值为 10，这种情况通常称为 10 折交叉检验（ten-fold cross-validation）。

8.6.6 标准文档集

与检索评价的情况类似，多个标准文档集制定出来，用于进行试验和分类技术对比。这些标准文档集的应用认为是近年来文本分类快速发展的主要原因之一。在下面的内容中，我们将介绍一些最常用的标准文档集。

1. Reuters-21578

Reuters-21578 语料库［337］是在分类实验中使用最为广泛的文档集。它是由路透社（Reuters）在 1987 年的新闻文章所组成的。该文档集被划分为多个与经济有关的类别（如兼并、收入等）。有一个叫做 ModApte 的标准划分，将 9603 篇文档用做训练，3299 篇文档用做测试，共有 90 个类别在训练集和测试集中同时出现。许多研究人员在他们的实验中只使用该文档集中 10 个最大的类别［183，520，838］。在训练集中，类别所占的比例从 1.88%～29.96%不等；在测试集中，这一数据为 1.7%～32.95%。

2. RCV：Reuters Corpus Volumes

Reuters Corpus Volumes 1（RCV1）是一个由超过 300 000 篇人工分类的新闻专线报道所组成的参考文档集，最近由路透社发布［1345］。所有的报道被组织成 103 个主题类别，人们希望能够用该集合在文本分类实验中代替前面提到的 Reuters-21578 文档集合，因为它相对于后者要大得多（大约 35 倍），而且非常可靠，也更加干净。RCV2［1346］是原始发布的文档集的一个改进版本，将其中的一些错误数据进行了纠正［1016］。

3. OHSUMED 参考文档集

OHSUMED 文档集［750］是 MEDLINE 数据库的一个子集，由美国国家医药图书馆（National Library of Medicine，NLM）维护的授权医疗文献所组成。MEDLINE 数据库在 2009 年有超过 1700 万篇文献，每篇文献都包括由人类标注的医学主题词（Medical Subject Headings，MeSH）。使用的医学主题词表包含了超过 17 000 个医学主题词。 329

OHSUMED 文档集由在 1987—1991 年期间，从 MEDLINE 数据库覆盖的 270 种学术期刊中选择出来的 348 566 篇医学文献所组成。文档集同样包含了由内科医生在照顾病人过程中产生的 101 个查询请求。对每个查询结果，有三种可能的相关性判断：绝对相关（definitely relevant）、可能相关（possibly relevant）、不相关（non-relevant）。

由于 MEDLINE 文献包含医学主题词，因此这些主题词可以解释为医学类别，因而 OHSUMED 文档集就非常适用于文本分类算法的评价。要下载该集合，读者可以登录站点 ftp://medir. ohsu. edu/pub/ohsumed。

4. 20 NewsGroups

另一个广泛使用的文档集合是 20 NewsGroups[402]。这是一个由大约 20 000 条发布在 Usenet 新闻组上的消息所组成的集合。Usenet 可以近似平均地分为 20 个不同的新闻组。消息的类别即是其所属的新闻组。

5. 其他文档

其他在文本分类文献中使用或者报告的文档集包括：WebKB 超文本集合 [401]、ACM-DL（ACM 数字图书馆的一个子集），维基百科（Wikipedia）以及某些 Web 目录，如 Yahoo！和 ODP。

8.7 类别组织——构建分类体系

标注能够为每个类别提供语义信息，非常有助于分类过程的进行。然而，即使所有类别都已经进行标注，我们仍然只有一个“扁平”的类别空间，因为还没有提供类别的组织结构信息。缺乏类别组织信息会导致在理解和推理问题上的诸多限制。对此一种解决方案是构建分类体系（参考 6.7.1 节）。例如图 8-21 对于我们在本章前面所使用的例子显示了针对酒店的分类体系。在这个例子中，泛化是基于酒店的位置来实现的，因此我们称该分类体系是地理参照的（geo-referenced）。

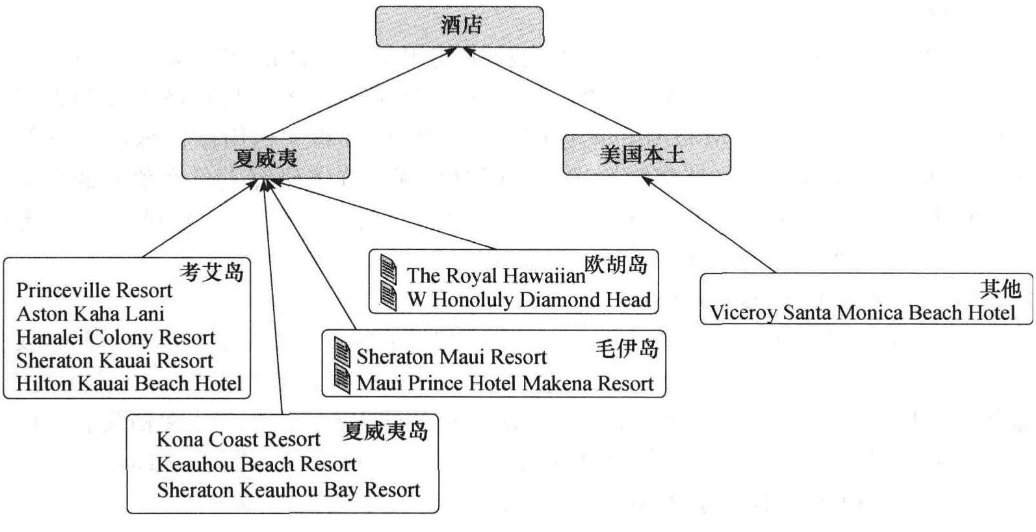


图 8-21 根据地理参照分类体系构建的夏威夷酒店网页组织形式

通常分类体系是通过人工或者非常复杂的半自动过程 [1001] 构建的，如图 8-22 所示。图中显示的五个步骤可以分解为以下更小的任务 [1159]：

1）构建一个能够表示领域知识的小文档集，通常包含 1000 篇左右的文档。

2）对文档进行分析，对其内容进行描述并对文档分组。

3）通过与领域专家进行沟通，明确需要为之构建分类体系的用户的信息需求。

4）确定并编制领域知识中的概念描述（用于表示类别）。 330

5）识别在领域知识中最常用的语言项，如名词和名词组。

6）使用识别出的索引项对领域知识中的概念进行标注（即标注类别）。

7）根据特化和泛化的关系，层次化地组织标注好的概念。

8）建立语言的等价关系，如同义词。

9）与用户一起测试并验证分类体系。

10）修正并最终确定分类体系。

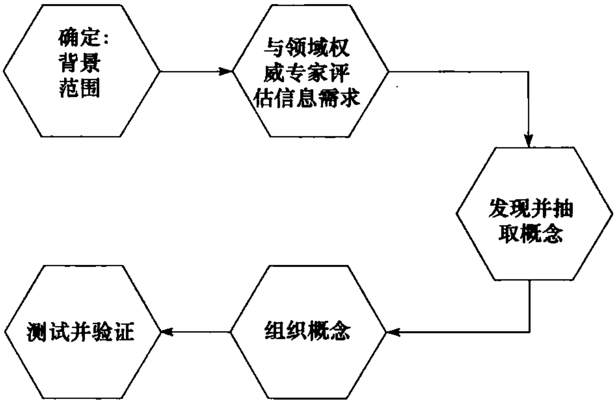


图 8-22 构建分类体系的过程 331

每个步骤都需要大量人工，如果分类体系需要构建 1000～2000 个概念，那么整个过程可能需要数个星期才能完成。而且，即使在无法预先知道有哪些类别的情况下（甚至并未打算构造分类体系的情况下），以上的过程也能够定义特定领域知识的类别集合。这是有用的，因为类别集合是将文本分类算法应用于未知的、新领域知识的先决条件。

尽管有人可能希望尝试自动构建一个分类体系，但以当前的发展水平来看，人工构建的分类体系会有更好的质量。最重要的是，人工构建的分类体系能更好地反映用户的信息需求（毕竟他们会根据自己的偏好进行输入）。分类体系的自动构建仍然是一个有待更多研究与发展的领域。

定义了某个知识领域的分类体系后，文档集合中的文档可以根据其概念（或者类别）来进行分类。举例来说，对于医学领域，给定如 UMLS 这样的医学分类体系后，医学文档集就可以分类到这个体系中，于是一系列所需要的不同类型的信息就可以直接给出答复：

- 从上个月开始，哪些是关于心肌手术技术的文档？
- 哪个概念（类别）发生了稀有事件（比如新文档的到达率突然改变）？
- 哪个是用户查询得最多的概念？
- 描述最流行的查询的概念模式是否与描述最近发表的文档的概念模式一致？

在将文档根据分类体系分类之后，可以快速地对许多文档集和查询流的特征进行检查和定量分析。分类体系可以用来构建某个给定知识领域的知识库，从而增强其功能性并且更好地获取领域中的语义信息。

尽管我们这里讨论的文本分类算法没有像文献［1032，1351］那样利用分类体系的层次结构而对算法进行调整，但这些算法仍然可以直接用于将文档分类到分类体系里所定义的类 332

别中。

8.8 趋势和研究问题

随着 Web 和其他数字存储形式所包含数据的激增，当前的分类技术面临的一个显著挑战是可伸缩性和可扩展性，以便能够处理日益增长的文档、越来越庞大的分类体系以及其固有的不平衡性与稀疏性。比如，在 2004 年的 Yahoo! 分类体系中，大约有 300 000 个类别，被组织成一个 16 层的层次结构 [1040]。而且类别的分布是非常不均匀的——在大约800 000 个页面中，拥有 100 个以上文档的类别不到 1%。处理这样的情况显然对于分类技术来说是巨大的挑战。使用链接信息进行 Web 文档分类的早期工作可以参考文献 [312，446]。

其他有待于更多研究的主题包括了层次分类和多标签分类。对于前者，分类器需要考虑类别之间的关系，相比于忽略层次结构信息和分别独立地处理每个类别的方法，它能获得更好的性能。类似地，处理多标签分类问题任务的常用方法是简单地对每个类别学习一个独立的二类分类器，而不是利用标签之间的依赖关系。尽管已经有了一些不局限于这些基本策略的成果 [521，1040，1634]，但仍然存在着巨大的改进空间。

另一个值得注意的问题是将一些当前的分类技术应用到某些新颖而有趣的领域中，如 1) *情感倾向分类*（sentiment classification），即判断对于某个事物的特定意见是正面的还是负面的 [1241]；2) *流派分类*（genre classification），即对特定文档所属的性质、类别和风格进行推理 [998]；3) *实体解析*（entity resolution）[695，1254，1255]，即判断两个实体引用是否指代真实世界中的同一个实体。将这些问题本身固有的一些性质与分类策略结合起来，可以获得比单纯使用分类技术更好的结果。另一个非常困难的任务是多媒体内容的分类，因为通过自动索引的方法获取富媒体的语义信息是很困难的。

另一个研究方向是*半监督分类技术*（semi-supervised classification technique），即在学习过程中同时利用未标记文档和一小部分标记文档 [1207]。比如可以通过将未标记文档加入到训练集中，从而使分类器对于分类的正确性有更高的置信度。这些方法的动机来自那些无法获得足够的训练数据，或者获得训练数据将会非常昂贵的情景。

另一个最新的研究方向是解决文档和类别*随时间演变*的问题，因为知识领域总是在不断地演化。我们需要制定合理的策略，能够较好地在选择大量训练文档建立分类模型与适应时间演变之间寻找一个合适的平衡点 [470，1160]。

最后，随着分类技术在文本上的成功应用，最新的研究趋势是将这些技术（和其他技术）应用到为文档检索任务学习排序函数上去（称为 Learning to Rank 问题）。这些年这股势头发展得十分迅猛，比如将 SVM 应用到点击流数据的偏好学习上 [843]。近年来，发表
333 了许多关于该问题的研究成果，比如在近些年的 SIGIR 会议中 [1308，1759，479，1632]。

8.9 文献讨论

两个最早将决策树应用到文本分类上的结果发表在 [1015，605] 中。在决策树中，处理缺失值或者未知值的问题在 [595] 中进行了讨论。决策树同样是集成学习中优先选择的方法之一，主要是 boosting 算法 [1433，1432]。

将 k 近邻应用到文本分类的经典文献是 [1095，1741]。关于 k 近邻更深层次的讨论和其中的一些问题可以参考文献 [712]。其他用于文本分类的延迟方法（以 k 近邻为基础）近来也受到关注，并取得了很好的结果 [1633]。

将 Rocchio 分类器的基本形式应用于文本分类的结果发表在 [797]。Schapire、Singer

和 Singhal 在 Singhal 早期工作 [1485] 的启发下，提出了基于查询区域的 Rocchio 分类器，并应用于文本分类任务 [1433]。

关于朴素贝叶斯在文本分类上的应用，有一个很好的讨论，发表在 [1014] （并在 [1446] 中进行了总结）。他们的工作同时也讨论了这项技术的未来发展方向，比如放宽独立性假设。对于朴素贝叶斯文本分类器的两种事件模型的比较发表在 [1105]，结果说明包含索引项频率的模型（称为多项模型，multinomial model）几乎总是会比不考虑这一信息的模型（称为伯努利模型，Bernoulli model）产生更好的结果。

有一本书包含了支持向量机的基本内容 [1628]。Joachims[838] 和 Dumais 等人 [520] 最早将支持向量机应用到文本分类中，而且在他们的工作中支持向量机都取得了比其他基线更好的结果。Yang 和 Liu[1743] 以及其他人同样证实了支持向量机可以取得当前最好的性能水平。将支持向量机实际应用于大规模数据集（比如文本文档集）的工作在 [839] 中进行了讨论。最近的一个重要结果发表在 [843]，它描述了一种能够在线性时间内训练线性支持向量机的方法，因此大大提高了其效率和对大规模数据集的可扩展性。多类支持向量机的构想发表在 [1680，436]，对方法所进行的比较发表在 [781]。

许多工作都报告了将集成学习分类器应用于文档分类的结果。Yang 等人 [1742] 使用了一种将三个不同类型的分类器（k 近邻、Rocchio 和基于语言模型的分类器）的归一化输出结果进行线性合并的方法。分类器的参数通过验证集进行调整，最后应用于测试集，产生了很好的结果。在文献 [979] 中，作者将三种不同的分类器进行了线性加权合并，其中权重是基于每个分类器在验证集上的性能进行调整的。他们同样测试了几种将两个分类器合并的方法（称为两路合并），以及一个三路合并方法。所有两路合并的方法都取得了比单一分类器更好的结果，而三路合并又获得了比所有两路合并更好的结果。不过他们使用的测试集非常小。

Li 和 Jain[1022] 使用了一种基于四个分类器（包括了 k 近邻、朴素贝叶斯和决策树）的集成学习方法，并使用了三种不同的合并方式：1）简单的多数投票；2）动态分类器选择（DCS）[1721，624]；3）自适应分类器合并（ACC)。在动态分类器选择（Dynamic Classifier Selection，DCS）方法中，集成分类器中的一个（对于与给定文档 d_j 最相似的文档有最好的相对分类性能的那一个）被选择用于对 d_j 进行分类。在自适应分类器合并（Adaptive Classifier Combination，ACC）方法中，每个分类器的预测结果通过加权和合并，其中权重为每个分类器的相对性能（对与文档 d_j 最相似的文档进行分类时）的函数。最后一种是最 334
好的合并方法。Lam 和 Lai 提出了一种略微不同的方法 [969]，他们使用训练集中类别的一些特殊特征（如训练样本个数、平均文档长度和平均索引项权重）学习这些特征与分类错误之间的关系。这样就能针对每个特定的类别选择不同的算法。在著名文档集 Reuters-21578（参考 8.6 节）上的实验证明，集成分类器与所有成分相比，在性能上都得到了提高。

最早将 boosting 应用于文本分类的工作发表在 [1433，1432]。这里的集成分类器由单层决策树组成，在树根部的测试是简单地检查某个索引项在文档中是否出现。boosting 方法的四个变体发表在 [1432]。这四种方法中的三种，我们称为 AdaBoost. MH。这一方法的目的是最小化分类错误（即海明损失，Hamming loss[1432]）。对于第四种，我们称为 AdaBoost. MR。这一方法的目的是最小化排序错误，即为多个标签确定其最正确的排序。Sebastiani等人 [1447] 扩展并改进了 AdaBoost. MH 的思路，其中的弱假设不再是一个学习器，而是一个集成分类器或者一组弱分类器。这样的方法可以提高算法的可用性和效率。无论是原始的版本，还是 Sebastiani 的改进版本，都可以通过连续属性的离散化进行扩展，

从而进一步改进该算法 [1172]。

两个将叠加技术 (stacking) 应用到文本特性描述的例子发表在 [249] 和 [183]。在第一个工作中，通过将多个对文档的不同结构成分和文档全文进行训练的分类器 (包括朴素贝叶斯和支持向量机) 的预测结果合并起来的方式，使用 stacking 来对半结构化的文档进行分类。线性 SVM 用于元分类。Bennett 等人 [183] 在分类器的预测结果之外，还使用了可靠性指标 (reliability indicator) ——即考虑分类器在不同情况下的性能因素，包括文档的信息量、分类器预测结果对于证据改变的敏感程度，以及某些关于预测结果的简单统计量 (如一致率)。他们使用了决策树和支持向量机作为元分类器。在这些研究中，集成分类器几乎在所有情形下都得到了有竞争力的结果 (大多数情况下取得了比单个成分更好的结果)。

已经存在了若干个用于构建决策树的软件包，如 ID3[1311]、C4.5[1313]、C5[1398]。SVMLight[839] 和 LibSVM[355] 是两种最常用的 SVM 工具包。SVMPerf 工具包 [840] 包含了在 [843] 中描述的线性方法的实现。最后，Bow[1104] 和 Weka[1707] 是两个非常流行的工具包，包含了许多本章介绍过的方法。

这些技术的大多数前沿进展 (或者新技术)，不限于文本分类，一般发表在机器学习和数据挖掘的会议上，如 International Conference on Machine Learning (ICML) 和 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)。将这些方法应用到文本上的实验，以及其他与文本分类有关的前沿工作，主要发表在 SIGIR
335 和 CIKM 会议上。

索引和搜索

——与 Gonzalo Navarro 合著

9.1 介绍

检索系统的主要目的是帮助用户找到他们感兴趣的信息。它的目标是提高*有效性*（effectiveness），即最大化用户的满意度与他们所付出的努力之间的比值。本章将关注检索的另一个方面，*效率*（efficiency）。我们将查看检索系统的一些实现方法，它们能够用尽可能少的计算资源（如 CPU 或耗时）、I/O 时间、网络带宽、内存空间、硬盘空间等来处理用户的查询。

尽管相比于有效性，效率可能只是次要考虑的问题，但它在检索系统的设计中几乎无法被忽略。以每次只服务一个人、仅管理 1000 本书的小型图书馆系统为例，在办公室的墙边堆满书架通常就足够容纳这么多书。现在假设每本书以电子形式占 700KB，那么所有的文本就占用 700MB。在这个文档集中简单地查找一个词，如果是在现代台式计算机上，使用公开的一些序列搜索工具，如 grep，而且假设整个文档集能够放在内存中，那么这个查询在不同机器上将花费 1～10 秒。考虑到现在适度规模的图书馆已经增长到 10 000 本书，存储为电子形式需要 7GB 的空间，因此不可能再完全放到内存中，那么查询将花费两三分钟。无论检索模型是多么有效，查询界面是多么别致，如果用户等待一个查询需要 2 分钟，那还是十分令人沮丧的。

这个例子说明，大多数检索系统在设计时必须考虑效率。而且，当我们转移到大规模应用时，相对于有效性，效率变得越来越重要。例如，在索引 TB 级别的数据并每秒服务数十万查询的 Web 搜索引擎中，效率关乎生存的重要性，因此只能使用最简单的提高有效性的技术。换句话说，为了能够运行更好的搜索算法，有必要实现更加复杂的技术来提高有效性。 337

正如之前所说的例子，序列搜索是搜索的最基本的形式。它不需要建立或者维护文本结构。但是，正如已经说明的，这通常只在有限的一部分情况当中才具有实用性。然而，我们稍后会展示它的相对重要性。

在大多数实际情况中，*索引*（index）是必须使用的。索引是建立在文本上的一种数据结构，用于加快搜索速度。维护和处理索引要比运行序列搜索复杂得多，但是在大多数情况下，这是能够得到可接受的响应时间的唯一办法。在使用索引的检索系统中，系统的效率可以用如下方式来衡量：

- **索引时间**：建立索引所需的时间。这里我们可以测量所有花费的时间，或者可以分开考虑 CPU 时间、I/O 时间等。通常来说，索引时间与文本大小是线性关系。
- **索引空间**：生成索引时所使用的空间。这个可以通过在建立索引时需要的最大空间来衡量。通常，它与被索引文本的大小是线性关系。
- **索引存储**：当索引生成以后，保存索引所需要的空间。目标是最后的索引存储空间应该是文本总大小的一小部分，而且它应该比在索引过程中需要的空间小得多。
- **查询时延**：从查询到达检索系统与答案生成之间的时间间隔。平均时间是更常用的

指标，但考虑一定百分比的查询（如90%的查询）的最大时间也可能是重要的。注意，我们并没有包括用户可能会体验到的任何其他时延，如在Web检索系统中，发出查询请求后与看到答案前的网络时延。

- **查询吞吐量**：每秒钟处理的平均查询数目。它直接根据查询时延来计算。

在附录A中，我们比较了几个开源的搜索软件包（搜索引擎），它们采用了很多这样的效率指标。

尽管这些是主要的性能指标，但还有其他一些重要的考量。当一个文本更新后，任何建立在它之上的索引也必须更新。如果文档集十分易变（也就是说，经常改变），那么这就不怎么实用了。现在的索引技术并不能很好地支持经常改变的文档集。相反，它们可以处理所谓的*半静态文档集*（semi-static collection），这些文档集以合理的固定时间间隔更新（如每天）。大多数实际的文档集，包括Web，实际上都是半静态的。比如说，尽管Web变化得十分快，但是搜索引擎的爬取相对较慢。为了维持新鲜度，常会使用增量索引。

对一个经常改变的文档集维持很高的新鲜度是可能的，除了在当前索引上检索外，还要
338 对最近一次索引之后获取的文档进行序列搜索。如果在连续两次索引更新之间增加到文档集中的新文本数目不是很大的话，那么这种方法是可行的。另一种解决方案是为新文本建立一个较小的索引，然后在每次查询时检索两个索引。

此外，当考虑索引方法时，需要把索引所占用的额外空间考虑进去。为了减小占用的空间，许多索引方案在内部工作时采用序列搜索。

我们首先讨论最常用的加快检索速度的索引技术：基本倒排索引。许多模型使用这种索引来实现检索，如简单的布尔模型，在传统检索系统中使用最多的排序模型——向量模型，还有经典的概率模型，以及大多数基于词频对文档排序的其他模型（见第3章）。

我们还要讨论用于全文检索的倒排索引。在这种情况下的任务是找到词语在文档集中的所有记录（occurence），这里的“记录”可以定义为词所出现的文档。它可能还需要确定词在文档中出现的位置。然后我们继续讨论如何用倒排索引处理更多复杂的搜索：复杂的模式、短语、布尔查询和结构化查询。对于复杂的模式，我们使用7.1.2节所讲的语言，该语言允许通配符、字符类（class of character），甚至正则表达式，匹配方式可以是精确的或者容错的。

我们接下来讨论让索引支持排序，这相当于根据检索结果的相关度对它们进行排序。因为对所有的答案都这么处理的代价比较大，所以我们专注于怎样获得最好的k个结果（k是一个合适的较小的值）。我们还会讨论索引压缩，介绍压缩的倒排索引如何能够正常运作，而且比正常的形式占更少的空间。

在倒排索引上解决复杂模式的查询涉及在文档集的词汇表上进行序列扫描。此外，序列搜索算法对于解决短语查询是必要的，它还可以在界面中加亮显示出现的记录。这就说明了序列搜索算法的重要性，即使是在索引检索中。序列搜索算法将在本章的后面介绍。

接下来，我们会描述后缀树和后缀数组。它们也是全文检索的索引，比倒排索引更加强大，也更难维护。例如，后缀数组能比倒排索引更快地搜索长短语。此外，能够在任何种类的文本上建立后缀数组，不仅仅是那些由词语组成的文本。这使得它适合处理像中文、日文和韩文这些很难分词的语言。甚至像芬兰语和德语这种黏着语在使用基于词的模型时也会有问题。而且，后缀数组能够用于其他没有词语的应用，如计算生物学和音乐检索。我们还会讨论压缩的后缀数组，它们比正常的形式占用更少的空间。

然后，我们简要地讨论签名文件（signature file）。尽管如今签名文件在大规模系统中

很显然不可能与倒排索引相抗衡，但在分布式应用中它仍可能是有用的，在这些应用中需要有一个快速小巧的过滤器来确定在远程的索引上是否存在一个潜在的匹配。

在此之后，我们描述一些序列搜索算法，从简单的字符串搜索到基于正则表达式和近似模式匹配的更复杂的搜索。我们还将讨论搜索和压缩之间的关系。这与 6.8 节有所联系，在 6.8 节中我们描述了压缩算法以及它们在搜索单个词模式中的应用。 339

我们还加入了介绍多维索引的一小部分内容，也就是说，从多个维度建立索引，而不仅仅是像文本的一个维度。尽管搜索多个维度的可扩展性仍然是个问题，但这还将继续是搜索多媒体内容的主要技术。本章的最后是索引和搜索文档集方面当前的趋势和研究热点。

本章自始至终假设读者熟悉基本的数据结构，如排序数组、二分查找树、B 树、散列表和 trie 树。因为 trie 树在本章中使用得较多，所以 9.4 节对其加以简单的介绍。

9.2　倒排索引

9.2.1　基本概念

倒排索引（也称为倒排文件）是一个为了索引文档集、加快搜索任务的面向词的方案。倒排索引结构由两个元素组成：**词汇表**（vocabulary）（或叫做词典）和**记录**（occurrence）。词汇表是指在文本中出现的所有不同词的集合。对于词汇表中的每个词，索引保存了所有包含这个词的文档。因为这个原因，它叫做倒排索引，因为我们能够用索引来重建文档。这是如今主要的索引结构，也是最古老的。

表示包含词汇表中每个词语的文档的最简单的方式是使用矩阵，其中的每个单元是词语在文档中出现的次数——即第 3 章介绍的项-文档矩阵。如图 9-1 所展示的项-文档矩阵，对应于图 3-6 的文档集。这种简单的表示十分快速，只要访问一次矩阵就可以知道文档是否包含某个词。如果使用布尔模型，我们甚至不需要知道词语在每篇文档中出现的次数。在这种情况下，表示索引项是否出现在文档中的布尔矩阵就足够了。

词汇表	n_i	d_1	d_2	d_3	d_4	倒排表形式的记录
to	2	4	2	–	–	[1,4],[2,2]
do	3	2	–	3	3	[1,2],[3,3],[4,3]
is	1	2	–	–	–	[1,2]
be	4	2	2	2	2	[1,2],[2,2],[3,2],[4,2]
or	1	–	1	–	–	[2,1]
not	1	–	1	–	–	[2,1]
I	2	–	2	2	–	[2,2],[3,2]
am	2	–	2	1	–	[2,2],[3,1]
what	1	–	1	–	–	[2,1]
think	1	–	–	1	–	[3,1]
therefore	1	–	–	1	–	[3,1]
da	1	–	–	–	3	[4,3]
let	1	–	–	–	2	[4,2]
it	1	–	–	–	2	[4,2]

图 9-1　对应图 3-6 所示文档集的基本倒排索引和项-文档矩阵。每个矩阵单元表示项在文档中的频率。注意，词汇表保存了项 k_i 所出现的文档的数目 n_i；在矩阵中，数字是 0 的单元用“-”替代以提高可读性。在右边，项-文档矩阵由倒排表所替代，它的单元是由包含该项的文档和对应的项频组成的

这个简单的解决方案的主要问题是它需要太多的空间（与文档数目和词汇表大小的乘积成正比）。因为这是个稀疏矩阵（大多数文档仅包含词汇表的一小部分），所以解决的办法就

是把一个文档列表与每个词联系起来。所有这些文档列表的集合叫做“记录”[⊖]。图 9-1 的右边是相同例子的倒排表。所使用的空间与文档中词语的记录成正比，这比文档的大小要小得多。

到目前为止，倒排索引还是用于定位词语出现的所有文档，而不是根据某些排序模型找到最可能相关的文档。为了定位文档，倒排表以文档顺序排序是有好处的，即使这个顺序实际上是任意的。例如，我们可以使用一个全局的排序方案，并以那样的顺序遍历文档（当排序变化时动态地维护这个遍历过程）。比如，我们可以使用一个基于 TF-IDF 的排序，把文档中所有不同词的值加起来，或者在网页上我们可以使用基于 Page-Rank 的排序（见 11.5.2 节）。

为了实现向量模型和第 3 章提到的其他一些模型，我们需要将倒排表的顺序变成按照频
340 率降序的方式。通常，我们可能以降序保存每个索引项-文档对的权重，而不是保存在每个文档中出现的次数。这个变种叫做面向排序的倒排索引，将在 9.2.4 节中介绍。

9.2.2 完全倒排索引

我们上面所讨论的基本索引并不适合回答短语或者邻近查询（proximity query），因为它不包含每个词在文档中出现的确切位置的信息。因此，我们需要把每个词在每篇文档中的位置加到索引中去。这个位置指向词或者字符。词位置（如位置 i 表示第 i 个词）简化了短语和邻近查询，而字符位置（如位置 i 表示第 i 个字符）便于直接得到匹配文本的位置，如用于展示文本片段（snippet）（文本片段是来自文档的一小段文本，包含全部或部分查询）。这样的倒排索引通常叫做*完全倒排索引*（full inverted index）。图 9-2 展示了一篇文本的情况，每个项的记录用它在文本中的字符位置来表示。

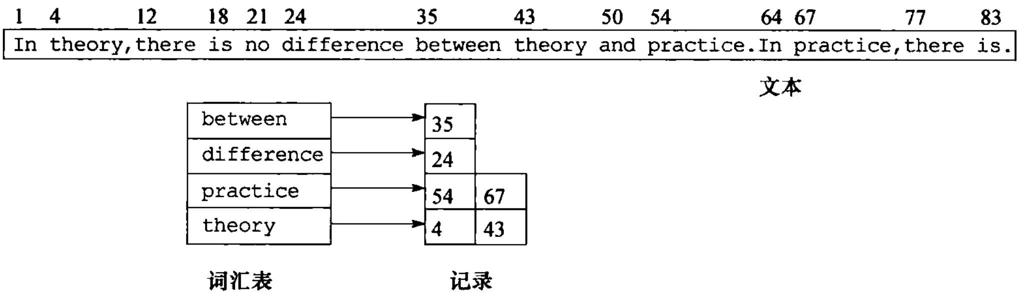


图 9-2 样例文本和在此之上建立的完全倒排索引。禁用词并未被索引。记录指向的是文本中的字符位置

在多个文档的情况下，我们需要为每个项-文档对保存一份记录列表。对图 9-1 中的那个例子，图 9-3 展示了这种情况，每个项的记录用文本中的词位置来表示。在每个列表中，第一个数字表示文档的编号，第二个数字是项在文档中出现的次数。保存一份完全倒排索引
341 所需要的空间将与所有记录的数目成正比，因此也就与文档的大小成正比。用词位置，而不是字符位置，能够将所使用的空间至少减少到 1/3，所以更推荐这种方法。去除禁用词同样能够减少一半的空间大小，因为这些词出现的次数大约占 50%，而去除禁用词通常不会改变排序，反而会提高检索的效果。在我们的例子中，禁用词的出现次数要超过所有记录的一半。6.6 节介绍的另外一些文本转换方法也可以减少记录的数目。

⊖ 在一些较老的论文中，你可能会发现用记录列表（posting list）和记录文件（posting file）来指这个集合。

词汇表	n_i	完全倒排表形式的记录
to	2	[1,4,[1,4,6,9]],[2,2,[1,5]]
do	3	[1,2,[2,10]],[3,3,[6,8,10]],[4,3,[1,2,3]]
is	1	[1,2,[3,8]]
be	4	[1,2,[5,7]],[2,2,[2,6]],[3,2,[7,9]],[4,2,[9,12]]
or	1	[2,1,[3]]
not	1	[2,1,[4]]
I	2	[2,2,[7,10]],[3,2,[1,4]]
am	2	[2,2,[8,11]],[3,1,[5]]
what	1	[2,1,[9]]
thank	1	[3,1,[2]]
therefore	1	[3,1,[3]]
da	1	[4,3,[4,5,6]]
let	1	[4,2,[7,10]]
it	1	[4,2,[8,11]]

图 9-3 全文倒排索引——索引项记录以在列表中的列表形式表示，位置用的是词位置

有些作者会区分倒排索引（或文件）和倒排表。在倒排索引中，列表中的每个元素指向一个文档或者文件名（最初的简单情况），而倒排表和这里的完全倒排索引的定义是一致的。我们倾向于不区分这两者，正如我们接下来会看到的，这是一个*寻址颗粒度*（addressing granularity）的问题，它的范围可以是从文本位置到逻辑块。而且我们更想称之为倒排索引，而不是倒排文件，因为文本并不一定只是一个文件。为了区分颗粒度，我们使用单词寻址或者文档寻址的倒排索引。注意，如果每个文档都保存在一个文件中，那么文件寻址就相当于文档寻址。通常，我们更喜欢使用文本的逻辑结构，而不是物理结构（如文件）。 342

词汇表所需要的空间是相当小的。根据 Heaps 法则（见 6.5.2 节），词汇表以 $O(n^\beta)$ 增长，这里的 n 是文档集的大小，β 是一个依赖于文档集的常数，实际应用中是 0.4～0.6。例如，在 TREC-3 文档集中，1GB 文本的词汇表只占用 5MB 的空间，即 $V \approx 158\sqrt{1\text{Gb}}$。这个空间可以通过 6.6 节介绍的词干提取或其他规范化技术进一步地减小。

记录需要更多的空间。因为在单词寻址索引中，每个出现在文本中的词都会被引用一次，这个额外的空间将是 $O(n)$ 级的，去掉禁用词后大概是文本大小的 40%，而如果也对禁用词索引的话将达到 80%。文件寻址索引相对小一些，因为对于一个词，每个出现它的文件只需要记录一次。根据文件大小的不同，文件寻址索引通常需要 20%～40%的文本大小的空间。注意，文件寻址的倒排索引对于找出单词出现的文件是方便的，但是无法解决短语或邻近查询，也不能在没有进行序列扫描的情况下定位记录在文本中的上下文。

块寻址（block addressing）技术被用来减小空间占用。所有文档的文本被分割成块，记录指向词语所出现的块（而不是确切的位置）。注意，简单倒排索引和把整个文档当做块来使用是等同的，通常为了方便管理块，块之间的边界应该与逻辑或者物理边界（如文档或者文件）相一致。

块寻址使得指针更小，因为它的数目比位置少。而且，所有在同一块中的词的记录都缩减成一个块引用（见图 9-4）。运用这种技术，索引只占文本大小的 5%。这样做的代价是，如果需要知道词语出现的确切位置（如对于邻近查询），那么必须在匹配的块上进行在线搜索。例如，有 256 个块的块寻址索引在数百兆字节的文本上就无法很好地工作了。

表 9-1 按比例展示了倒排索引在不同的文本大小下所占用的空间，分为包含和不包含禁用词两种情况。完全倒排表示倒排所有词（在表中记为“按词寻址”），并且保存它们的确切位置，每个指针 4 字节。文档寻址索引假设我们指向的文档是 10Kb 大小（每个指针所需要 343

的字节数依文本大小而定，如 1、2 或 3 字节)。块寻址索引假设我们使用 256 字节或者 64 000字节的块（每个指针 1 或者 2 字节)，它与文本的大小无关。通过压缩，指针所占的空间可以大为减少。我们假设，45%的词是禁用词，每 11.5 个字符就有一个非禁用词。我们对词汇表的估计基于 Heaps 法则，其中的参数是 $V=30n^{0.5}$。所有这些决策都来自经验，并已经实验验证。

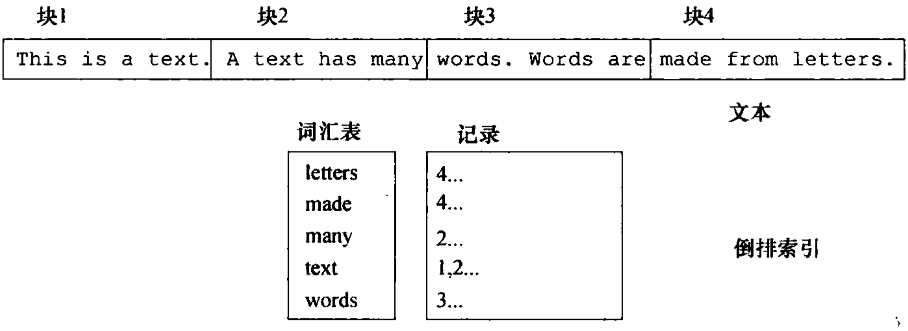


图 9-4 分成 4 个块的样例文本，以及在此之上建立的用块寻址的倒排索引。记录表示块号。注意词“words”的两个记录合并为一个块引用

表 9-1 倒排索引的大小占整个文档集大小的近似百分比。考虑了 4 种颗粒度和 3 个文档集。对于每个文档集，左栏是没索引禁用词的情况，而右栏是所有词都被索引的情况

索引颗粒度	单个文件（1MB)		小文档集（200MB)		中等文档集（2GB)	
按词寻址	45%	73%	36%	64%	35%	63%
按文档寻址	19%	26%	18%	32%	26%	47%
按 64K 大小的块寻址	27%	41%	18%	32%	5%	9%
按 256K 大小的块寻址	18%	25%	1.7%	2.4%	0.5%	0.7%

块的大小可以是固定的（意味着在文本数据库上使用逻辑块结构)，或者它们可以通过以天然的分割方式将文档集分成文件、文档、网页或其他方式加以定义。分割成固定大小的块能够在检索时提高效率，也就是说，块的大小波动越大，（在检索时）平均需要按序遍历的文本越多。这是因为较大的块更多地匹配查询，遍历的代价也更高。

或者，用自然的分割方式就可以不需要在线遍历。比如，如果每个检索单元是一个块，而且并不需要确切的匹配位置，那么对于单个词查询就没必要遍历所有的文本，因为知道是哪个检索单元已经足够了。但是，在另一方面，如果许多检索单元都放在同一个块中，那么必须遍历这个块来决定获取哪个检索单元。

需要注意的是：为了使用块寻址，文本必须在搜索时就已经准备好了。而不是像远程文本（如 Web 搜索引擎)，或者挂载 CD-ROM 中的文本的那些情况。同样，如果块就是检索
344 单元，那么不需要知道确切位置的受限检索也可以解决问题。

9.2.3 搜索

接下来，我们假设已经有了完全倒排索引，其中的倒排表按照文档标识符排序。因而，下面所说的算法不仅可以用于布尔模型，而且当存在基于全局排序的标识符方案时，可以用于返回前 k 个排序文档的模型。然后，我们讨论用于其他排序方案的倒排索引。

1. 单个词查询

检索的最简单（也是最常见的）类型就是查询单个词的记录。这对倒排索引来说是直截了当的：在词汇表中查找这个词，然后得到对应的记录列表，再以文本位置升序的形式直接提交。

词汇表搜索可以用任何合适的数据结构来实现，如散列、trie 树，或者 B 树。前两个的查询复杂度是 $O(m)$（与文本大小无关），这里的 m 是查询的长度。采用简单的方法，如以词典顺序保存词汇表中的词，它占用的空间更少，而且性能也可以接受，因为如果使用二分查找，查询词可以通过 $O(\log n)$ 次字符串比较找到，这里的 n 是词汇表大小。我们注意到，词汇表在大多数情况下足够小，能够存储在内存中。相反，记录表通常要从磁盘上获取，但也可以将一部分存储在内存中（如 11.4.3 节所讲的缓存倒排表）。

如果索引并没有保存足够的信息来提供期望精度级的记录（也就是说，索引地址块或地址文件，但是需要词的精确位置），那么还需要在记录表中得到的单元上进行序列搜索。序列搜索算法将在 9.5 节讨论。

2. 多个词查询

如果查询多于一个词，那么需要考虑两种情况：合取（AND 操作）和析取（OR 操作）查询。也就是说，布尔查询的两个特定例子。由于文档集的大小，第一种情况在 Web 中比较常见。

合取查询表示检索查询中的所有词，对每个词获取一个倒排表。接下来，对所有倒排表求交集（intersect）来得到包含所有这些词的文档。有多个求交集算法来实现这个目的，它们取决于列表是如何存储的，因为它们可以是连续的，也可以是分段的。假设列表是连续存储的，并以文档顺序排列。最简单也是用得最多的启发式方法 [491] 是，按照长度对列表排序，并从最短的两个列表开始求交集，然后对结果和下一个最短的列表求交集，以此类推。其出发点是，在某些点，交集变成了空集，这样我们就不用再继续做下去了。注意，交集为空出现的概率应该在短列表中比较高。对多于两个词的查询，这个算法在最坏情况下是超线性的，但平均来说是次线性的。

析取查询表示检索查询中所有的词，对每个词收集一个倒排表。因此，在所有这些情况中，列表必须合并，使得它们以文档或者文本位置顺序升序排列。同样，有可能必须进行一些序列搜索来得到确切的位置。需要注意的是，推荐先合并记录列表，然后进行序列遍历，因为这样可以避免一个文档出现在多个列表中的情况下多次遍历文本。接下来，我们讨论简单合并算法的一些变种。 345

3. 列表求交集

因为在倒排索引中最耗费时间的操作是合并记录列表或者对之求交集，所以优化这些操作是十分重要的。考虑一对需要求交集的大小分别是 m 和 n 的列表，如果 m 比 n 小得多，那么最好在较大的列表上做 m 次二分查找，这个算法的复杂度是 $O(m\lg n)$。因此，如果 m 是 $O(n/\lg n)$ 级的，那么这个算法比线性合并算法更好，后者的复杂度是 $O(n+m)$。注意，每次二分查找可以在较大列表中上次二分查找停止时的位置到最右位置之间进行。

对于这种情况，另一种可能的方法是使用*成倍扩展搜索*（doubling search）[490]，它与二分搜索有同样的复杂度，但是不需要一直把整个列表放在内存中。它的思想是，把较小列表中的元素和较长列表中位置 $2^i (i \geq 0)$ 上的元素进行比较，直到我们发现所查找的文档在上两次查看的位置之间。我们继续递归地对较短列表中的下一个文档从较长列表中的下一个位置开始成倍搜索。然而，如果 m 和 n 是可比较的（数量级），Baeza-Yates[87] 设计了一个成倍二分搜索算法，当交集接近为空（$O(\log n)$）时，它的速度十分快，而且需要的平均比较次数少于 $m+n$[124]。实际上，前面两个算法的平均复杂度都是 $O(m\log(n/m))$。最近的一篇论文对这些算法和其他一些算法进行了彻底的比较，表明最好的算法也依赖于数据的分布 [143]。事实上，不同问题的困难可能是很不一样的，求交集的复杂度源自于此 [142]。然而，另外两篇论文表

明，对于压缩的倒排表，最好的算法可能是很不一样的 [1420，463]。

在列表大小不定的情况下，这些算法可以逐一使用。当列表多于两个时，根据列表的大小会有多个可能的启发式方法。对于三个列表的情况，最好的方案是先对两个最短的列表求交集，然后将结果与最长的列表再求交集。对于四个或者更长的情况，启发式的方法将依赖于部分结果，因此必须是自适应的。通常，建立一个平衡的合并树，在结束之前避免长列表效果会比较好。另一方面，实际上我们不会有多于 6～8 个列表的情况。因此，如果对两个最短的列表求交集后得到了一个十分小的结果，那么可能最好把它再与下一个最小的列表求交集，以此类推。

4. 更加复杂的查询

前缀查询和范围查询基本上就是析取查询。前缀查询和范围查询都表示词汇表中词典顺序上的一段区间，因此它们可以通过二分搜索、trie 树或者 B 树来高效地解决，但散列不
346 行。在这些查询中，通常会有多个词符合查询模式，因此最终我们同样会有多个倒排表，然后就可以用之前的算法了。

对于复杂的查询模式，如正则表达式或近似搜索，在词汇表上建立的数据结构用处很小（尽管可以在 trie 树上做回溯，见 9.4.3 节）。解决方案则是用 9.5 节所给出的算法，顺序地遍历词汇表来找出所有满足查询模式的词。对应的记录列表同样必须合并起来。

这样一个顺序遍历的代价并不会太高，因为它只在词汇表上进行，词汇表的大小大概与文本大小的平方根成比例。实际上，顺序遍历词汇表会花费零点几秒的时间。注意，这里的顺序遍历有一点特别，因为我们需要匹配词汇表中的整个词，而不是任何子串。在压缩文本上的模式匹配同样也使用这种词汇表扫描策略。

同样需要注意的是，这个方法只解决了模式必须匹配单个词的情况。例如，如果我们查找"**shallow**"，并允许有一个错误，那么我们会找到"**shalow**"的记录，因为它匹配查询的整个词。然而，如果文本错误地把词语分割了，如"**shall ow**"，或者与其他词连在一起了，如"**shallowwater**"，那么上面的机制就无法找出这些只有一个错误的情况了。只有序列搜索（见 9.5 节）或者后缀树和后缀数组（见 9.4 节）能够找到这些记录。

5. 短语和邻近搜索

这里我们讨论短语和邻近搜索（或叫做上下文搜索）如何在倒排索引上进行。这些搜索更加复杂，因为必须对每个词搜索，然后必须处理它们的记录列表来获取答案。

上下文搜索更难用倒排索引解决。每个元素必须单独搜索，并对每个元素生成一个按位置递增排序的列表。然后，对所有元素的列表进行同步遍历，找出所有词连续出现（短语）或者出现得足够近（邻近）的位置。这里，我们只需改变交集的定义就可以应用前面所讲的求交集算法。例如，在记录词位置的情况下，对于短语搜索，如果我们搜索短语"**a b c**"，那么我们需要在 **a**、**b** 和 **c** 的记录列表中分别找到连续的位置 i、$i+1$ 和 $i+2$。如果我们需要找出两个词 **a**、**b** 最多间隔 k 个词的情况，那么我们需要在 **a** 和 **b** 的记录列表中分别找到位置 i 和 j，并满足 $|i-j| \leqslant k+1$。

如果索引保存字符位置，那么短语查询就无法忽略分隔字符，同时邻近程度必须以字符距离来定义。

短语搜索的另一个解决方案是对两个词的短语建索引，并在词对上使用相似度算法。这个方法的主要缺点是词汇表会非线性地增长；尽管倒排表更短，但索引的总大小会增加 50%或更多。一种可能的方案是只索引查询中热门的词对，在短语搜索的索引空间和效率之
347 间取得平衡。

6. 布尔查询

我们现在介绍一些通用的集合操作算法。这些算法在操作结果集合时使用，如一般布尔查询中的情况。在 7.1.1 节描述过布尔查询，并定义了*查询句法树*（query syntax tree）。

一旦查询句法树的叶子确定了（用算法找出包含给定基本查询的文档），通过组合操作符就可以找到用户有可能感兴趣的文档（根据用户提交的查询）。一般来说，检索过程包括三个步骤：第一个步骤决定匹配哪些文档；第二个步骤决定匹配文档的相关程度，从而以合适的方式展示给用户；第三个步骤得到匹配的确切位置，使得如果需要的话，在浏览时能突出显示它们。

这个方案避免对不包含（与查询的）匹配（第一个步骤）或最后并不会展示出来（第二个步骤）的文档做一些无用功。然而，如果做一些额外的操作开销并不大，有些步骤是可以合并的。在某些场景中，有些步骤可能并不存在。

一旦查询句法树的叶子找到了不同类别的文档集合，那么树的内部结点对这些集合进一步进行操作。可以使用一些代数方法来优化这棵树，例如恒等式 a OR(a AND b)$=a$，例如，公共的子表达式，但是我们这里并不涉及这些问题。

因为所有的操作符都需要与在两个操作数位置上的文档配对，所以一个较好的方法是保持集合有序，这样像交集和并集这样的操作就能够顺序地处理两个倒排表，而且最后产生一个有序的倒排表。另外也可能会有其他不包含匹配文档列表的集合表示形式（如位向量）。

在这样的方案下，可以以*完全*（full）或者*惰性*（lazy）的形式对语法树求值。在完全求值的形式中，首先完全得到两个操作数，然后再产生完整的结果。在惰性求值的形式中，只有当需要的时候才产生结果，并且为了得到这个结果，两个操作数会递归地调用一些数据。

完全求值使得某些优化能够执行，因为事先知道结果的大小（可以用在长列表中二分搜索短列表中元素的方法将短列表合并为一个长列表）。在另一方面，惰性求值使得应用能够控制何时去获取新的结果，而不必为了获取结果而阻塞很长时间。混合方案也同样是有可能的，如一次获取所有的叶子结点，然后以惰性的形式来处理。这么做也许是有用的，如为了实现一些优化或者为了确保所有对索引的访问都是顺序的（因此减少磁盘寻道时间）。图 9-5展示了这种情况。

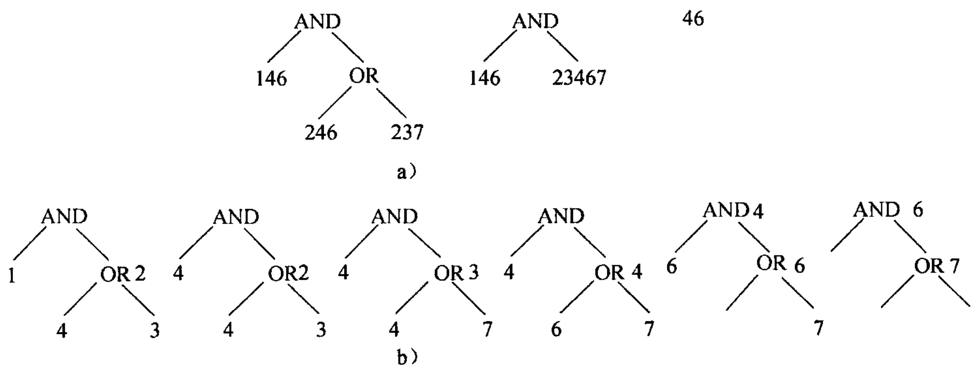


图 9-5 处理查询句法树的内部结点。a）中使用了完全求值；b）中详细地展示了惰性求值

解决这些查询类型的复杂度，除了获取叶子结点上的结果所需的开销外，通常与所有中间结果的总大小成线性关系。这也是为什么当有大量中间结果时，这个时间是起主导作用的。当最终结果集较小时这一点对用户来说更明显。

9.2.4 排序

本节讨论当有了按权重排序的倒排表后，怎样找到最佳的文档。一个重要的目标是，我
348 们并不想找到所有的答案，因为这太耗时间了。我们想找到的是排名前 k 个的文档，并把它们返回给用户。如果我们处理单个词查询，这个答案是简单的，因为倒排表已经按照期望的方式排好序，所以我们只需从倒排表中返回前 k 个文档就行了。

对于其他查询，我们需要合并倒排表。因为它们并未按文档编号排序，不能使用顺序合并实现有效的求交集操作，所以我们需要使用不同的算法。主要思想是计算每篇文档的排序，使其顺序接近于最后的顺序。最好的解释方法就是举个例子。考虑图 3-6 中的文档集和表 3-3 中根据 TF-IDF 权重排序的倒排表。假设现在我们搜索析取查询 **to do**，直接的方案就是如表 3-8 所示，计算查询与所有文档之间的相似度。然而，正如我们已经说过的那样，当文档数目十分大时，其代价会非常高。因此，我们要做的是先计算排名可能较高的那部分文档。因为我们的文档集很小，所以假设我们只对排名最高的文档感兴趣。对此，我们将维护一个排在前两名的文档集合，以便最后能得到最佳的文档。因为我们试图最大化 TF-IDF 的乘积，我们将使用下列启发式方法：1）以 IDF 顺序处理查询项；2）每个查询项以 TF 顺序处理。

在我们的例子中，从 IDF 值为 1 的 **to** 的查询项开始，从它的倒排索引中选择前两名的文档：$d1$ 和 $d2$，并计算这些文档与查询 **to** 之间的部分相似度。这是初始的候选文档集。然后继续检查查询项 **to** 的倒排表。现在，我们看下一个词 **do**，并以权重顺序查看它的倒排表。于是从 $d3$ 开始，计算这个文档与查询 **do** 之间的部分相似度。这个值比前两个候选文档的任何一个都要小，所以不用再继续检查这个倒排表了，因为倒排表中的其他文档也是这样的情况。然后，排名最高的文档就是前两个候选文档中最佳的一个，即 $d1$。注意，尽管我们没有计算整体相似度，但是这个结果与表 3-3 是一样的。

现在我们可以阐述通用的算法，它是 Persin 算法［1257，1258］的一个变种。我们使用一个包含 C 个候选文档的优先队列 P，将用这些文档计算部分相似度。然后按权重的降序顺序处理查询项，并对每个项计算阈值 t_{add}。这个阈值是指能够加入到部分相似度计算队列
349 的最小值。否则，我们可以忽略倒排索引表的剩下部分。这个算法的伪代码见图 9-6，其中

```
Ranking-in-the-vector-model( query terms t )

(1)  Create P as C-candidate similarities initialized to (P_d, P_w) = (0,0)
(2)  Sort the query terms t by decreasing weight
(3)  c ← 1.
(4)  for each sorted term t in the query do {
(5)    Compute the value of the threshold t_add.
(6)    Retrieve the inverted list for t, L_t.
(7)    for each document d in L_t do {
(8)      if w_{d,t} < t_add then break
(9)      psim ← w_{d,t} × w_{q,t}/W_d.
(10)     if d ∈ P_d(i) then P_w(i) ← P_w(i) + psim
(11)     elif psim > min_j(P_w(j)) then n ← min_j(P_w(j))
(12)     elif c ≤ C then {
(13)       n ← c
(14)       c ← c + 1
           }
(15)     if n ≤ C then P(n) ← (d, psim)
     }}
(16) return the top-k documents according to P_w
```

图 9-6 Persin 算法变种的伪代码，在向量模型中使用部分求值方法来计算排序的答案

主要有两种情况：如果文档已经在候选列表中，那么我们加一个新项（的相似度）到它的相似度上；否则，如果候选文档少于 C 个，或者它的相似度大于候选集中的最小相似度，那么我们初始化新文档的相似度。原算法讨论了如何设置阈值 t_{add} 的值，但是一个简单的方法是使用与当前候选文档的最小距离，对于每个项它在 $O(C)$ 时间内就能计算得到。原算法还使用了另一个阈值来添加候选文档（即没有固定的最大值），这使得能够在最后用文档长度 W_d 来归一化。我们的选择是设定候选文档的最大数目 C，它是 k 的一个函数（如 $C=10k$），这简化了算法，但是在循环中增加了一次除法。

这个算法的计算可以通过在索引中保存整数而不是实数来优化。这些整数可能会降低精度（如最大位数），尽管计算结果可能是近似值，但是它对最终排序引入的误差很小。正如 Ahn 等人 [54, 55] 在他们的按影响排序的索引中所展示的那样，在权重上引入文档长度也能够加速相似度计算。 350

9.2.5 构建

1. 内部算法

当我们可以把文本和索引保存在内存中时，建立和维护全文倒排索引是一个相对简单和低代价的任务。用于保存词汇表的动态数据结构（如 B 树、散列表）在创建时是空的。然后扫描文本，并在词汇表中依次查找每个词。如果是个新词，它就被插入到词汇表中。一旦该词已经在词汇表中，这个搜索就返回这个词的标识符，它是指向词汇表条目的指针。

除了词汇表，还会分配一个很大的数组，其中保存了文本中每个连续词语的标识符以及它在文本中的位置。一旦文本词语的序列转化成大数组中 **“标识符：位置”** 对的序列，这个数组按照标识符稳定排序（一种排序是稳定的，指的是值相等的键按照原始的位置排序）。在这样排序后，相同的标识符聚合在一起，在大数组中形成了连续的一段区间，其中位置域是升序的。通过使用每个区间的标识符，我们可以令词汇表中的词指向区间的第一个位置。然后构建过程就完成了，如图 9-7 所示。

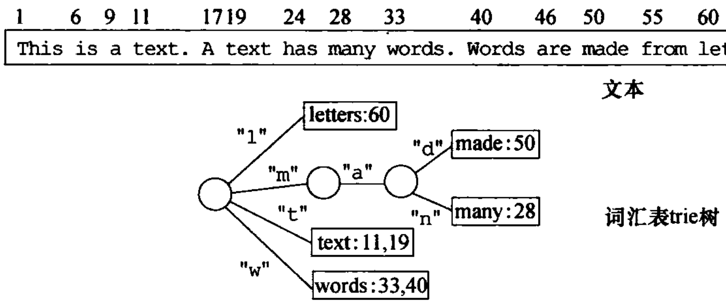


图 9-7 用递增算法为样例文本建立全文倒排索引

根据标识符的属性，我们有可能使用线性时间的桶排序。如果桶排序可行，这个过程将需要 $O(n)$ 时间；否则，时间将是 $O(n\log n)$，比桶排序算法慢。这个算法如图 9-8 所示。

避免排序的一个选择是从一开始就分割倒排索引。在这种情况下，词汇表中的每个词将保持一个指向它自己的记录数组（列表）的指针，刚开始是空的。然后，当在词汇表中找到文本中的词后，它的位置被插到列表的最后。当文本扫描结束时，记录列表就已经得到了。在倒排表支持排序的情况下，我们不能简单地在列表的最后添加；我们必须将新项插入到列表合适的位置，并更新对应文档的权重。这就意味着用于支持权重排序的列表的数据结构需要能在任何位置上更新。 351

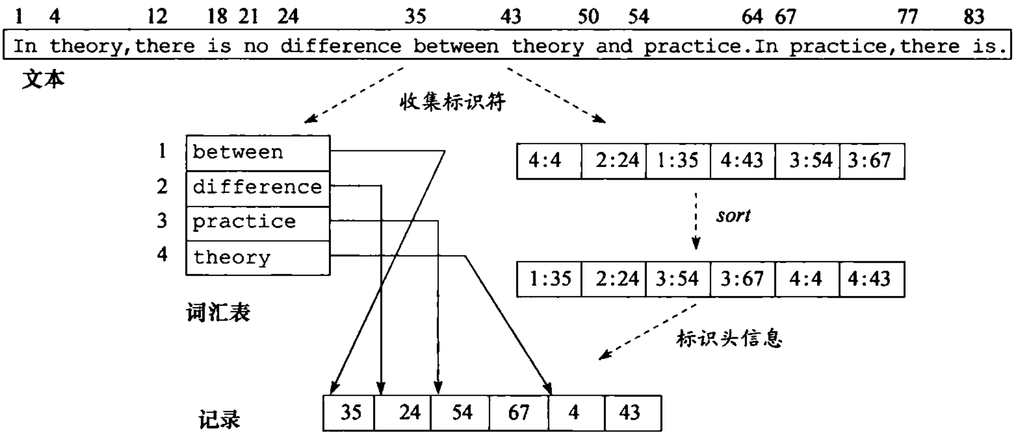


图 9-8 用排序算法对样例文本建立全文倒排索引

一个不小的问题是，如何为许多记录列表分配内存。每个元素都单独分配内存的典型列表浪费太多内存。相反，一个更可取的方案是，分配一个块列表，然后每个块中保存一些条目。但是，根据齐夫法则（见 6.5.2 节），许多词会有很短的列表，而少数词会有很长的列表。所以，很难选择块的大小，因为很小的块会在长列表中浪费很多空间，而大的块又会对不常用的词产生许多几乎为空的块。一个好的方案是，使用大小可变的块，这样对记录列表分配的第一个块是短的，随着越来越多的记录出现，更长的块会分配给它（因为这说明这个词是频繁出现的）。分配空间的一个方法是每次对列表的大小翻倍。这保证了在算法运行时内存分配操作造成的影响是 $O(\log n)$，而不是 $O(n)$。但在另一方面，我们可能会浪费更多的空间。

避免这个分配问题的一个方法是，先遍历一遍整个文本，计算所有词语的频率，这样就可以根据它们最终的大小来分配数组，然后再在第二遍时填充数组。然而，遍历两遍文本的代价比前面介绍过的两种技术都要高。但是需要注意的是，有时遍历两遍文本是无论怎样都要执行的操作，如对半静态文本进行压缩（见 6.8 节）。

一旦这个过程用任何一种方法完成了，词汇表就写成磁盘文件，而记录列表写成另一个文件。正如之前提到的，有时把第二个文件叫做*记录文件*（posting file）。对于每一个词，词汇表中包含了一个指向倒排索引中词记录开始位置的指针。这使得词汇表在大多数情况下能够在搜索时一直留在内存中。而且，只需很少或者不需要额外开销，一个词的记录个数也可能通过词汇表立刻知道。

总之，如果我们有足够的内存，那么递增算法通常是更好的，因为我们只需要遍历一遍
352 文本；否则，遍历两遍的算法仍然是合理的，因为它顺序地进行读取。

2. 外部算法

先前的构建方法只有当记录列表能够放在内存中时才有效。但在实际中很少是这样的情况，所以我们必须扩展这个技术来处理更大的文本集。所有这些算法都可以通过使用它们直到内存耗尽来进行扩展。这时，将当前已有的部分索引 I_i 写到磁盘上，并从内存中擦除，然后对剩余文本继续进行索引。

当所有文本都处理完以后，在磁盘上存在一定数目的部分索引 I_i。然后以一种层次的方式合并这些索引：索引 I_1 和 I_2 合并得到 $I_{1..2}$；I_3 和 I_4 产生 $I_{3..4}$，以此类推。这样生成的部分索引的大小大概是原来索引的两倍。当在这一层的所有索引都以这种方式合并起来后，在接下来一层上继续进行合并；将索引 $I_{1..2}$ 和索引 $I_{3..4}$ 合并，形成 $I_{1..4}$。这个过程持续到只剩

下一个包含了整个文本的索引。另外一些合并顺序也是可以的，如图 9-9 所示。

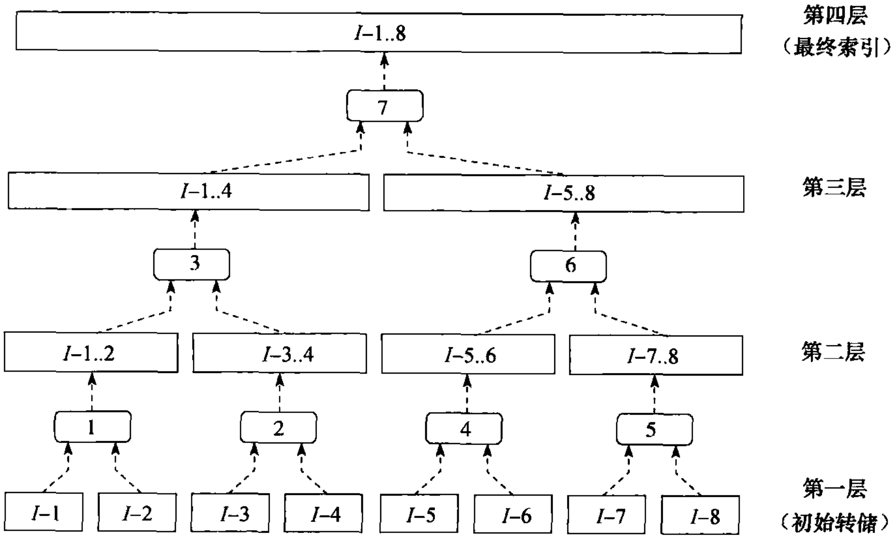


图 9-9 以二路的方式合并部分索引。矩形代表部分索引，而圆角矩形代表合并操作。合并操作中的数字展示了一个可能的合并顺序

合并两个索引首先是合并词汇表，其次当词语出现在两个索引中时，将两个列表中的记录合并。在构建过程中，编号小的索引中的记录先于编号大的索引，因此仅仅需要连接这些列表。在带权重的列表的情况下，我们需要合并它们。尽管如此，在这两种情况中，这都是一个十分快的线性时间过程。总的来说，如果 M 是可用的内存量，我们产生 $O(n/M)$ 个部分索引，合并它们需要的 I/O 代价是 $O(n\log(n/M))$，因为总共有 $\log_2(n/M)$ 个合并层，同时从一层移到接下来一层的 I/O 代价是 $O(n)$。如果我们对于每个小索引的构建都采用基于排序的方法，那么 CPU 的代价是每个索引 $O(M\log M)$，总共是 $O(n\log n)$ CPU 时间。如果我们对小索引使用 trie 结构或者散列方法，那么时间代价是每个小索引 $O(M)$，总共 $O(n\log(n/M))$。 353

每次可以合并多于两个索引。尽管这并不改变 CPU 复杂度，但是它提高 I/O 效率，因为有更少的合并层。另一方面，每个需要合并的部分索引的内存缓冲区将变得更小，因此会执行更多的磁盘寻道。实际上，一次合并多达 20 个部分索引也是较好的想法。在这种情况下，复杂度改进了 $O(\log R)$ 因子，这里的 R 是合并的基数。注意，如果使用更多的磁盘，性能会显著地提高，因为多个寻道过程可以同时进行。

通常，建议一旦文件可用时就合并它们（如图 9-9 所示），因为在不同的词汇表中重复的词合并成了一个，所以合并文件的词汇表比原来（部分索引的）词汇表的总和要小。另一方面，记录表上也没有了冗余。注意，词汇表可能会是较小的部分索引的重要部分。

如果使用文件寻址或者块寻址，这个算法只需略微改变。索引维护的代价也较低。假设一个大小为 n' 的新文本加入到数据库中，如对部分索引所做的那样，就会建立这个文本的倒排索引，然后与原有索引合并。这会花费 $O(n+n'\log(n'/M))$ 的 I/O 时间。删除文本可以通过如下方式实现：以 $O(n)$ 的复杂度遍历索引，删除那些指向需要删除文本区域的记录（同时，如果过程中它们的记录列表变为空，那么删除这些词）。更新文档可以通过删除旧文档并加入新文档来简单地处理。

总的来说，维护一个倒排索引可以用三种不同的方法来实现：

1）**重建**。如果文本并不是那么大，那么重建索引是最简单的方法。

2）**增量更新**。我们可以在搜索的时候分摊更新的代价，即我们只在需要的时候改变倒排索引。

3）**间歇合并**。如之前的算法所述，索引新文档，将得到的部分索引和较大的索引合并。一般来说，这是最好的方案。

在所有这些情况下，我们需要关心在索引更新时发生了什么变化。如果保留索引的一个备份，我们可以继续搜索而不发生问题。通常情况下，我们需要控制对索引的访问，以保持它的一致性。所以，如果有了更新，我们需要先收集它们，供将来的更新阶段来处理。在更新步骤中，我们可以有两个倒排索引：主要的一个和包含所有更新的较小的一个。在这种情况下，搜索必须在这两个索引上都进行，并合并结果。

9.2.6 压缩的倒排索引

在 6.8 节，我们已经叙述过适合文本数据库的压缩机制，那时只关注于减少空间。在倒排索引的情况中，可以将索引压缩和文本压缩毫无问题地结合起来。事实上，在之前所提到
354 的所有构建算法中，都可以加入压缩，作为最后一步。

我们从如何压缩全文倒排索引开始讨论（见 9.2.2 节），然后考虑用于排序搜索的索引（见 9.2.4 节）。通常，可以通过按每次压缩一个记录表的方式压缩记录文件，以极大地减小倒排索引的大小。如果我们想最大化内存中存储的索引量，这就十分重要。

在全文倒排索引中，每个记录表中的文本位置或文件标识符的列表都是以升序排序的。因此，它可以表示为相邻数字间的间距（gap）的序列。因为在许多情况下，记录列表是从开头开始顺序处理的，所以原来的文档编号可以很方便地通过加上这些间距计算出来。

通过观察知道常见词的间距较小，而较不常见词的间距较大，因此我们可以对小的值用较短的编码来实现压缩。一个可能的编码方案是*一元码*（unary code），这里整数 $x>0$ 的编码是（$x-1$）个 1 后跟一个 0，所以整数 3 的编码是 110。表 9-2 的第二列展示了整数 1～10 的一元码。

表 9-2 整数编码的例子

间距 x	一元编码	Elias-γ 编码	Elias-δ 编码	Golomb 编码（$b=3$）
1	0	0	0	00
2	10	100	1000	010
3	110	101	1001	011
4	1110	11000	10100	100
5	11110	11001	10101	1010
6	111110	11010	10110	1011
7	1111110	11011	10111	1100
8	11111110	1110000	11000000	11010
9	111111110	1110001	11000001	11011
10	1111111110	1110010	11000010	11100

Elias 提出了另外两种变长的整数编码方案。一个是 Elias-γ 编码，它通过连接两个部分来表示数字 $x>0$：1）对于 $1+\lfloor \log_2 x \rfloor$ 的一元码（它表示用多少位来代表 x）；2）二进制形式表示的 $x-2^{\lfloor \log_2 x \rfloor}$，总共 $\lfloor \log_2 x \rfloor$ 位的编码（这是 x 不带最高位的二进制形式，因为最高位总是 1）。对于 $x=5$（二进制是 101），我们知道它的长度是 $1+\lfloor \log_2 x \rfloor=3$，同时 $x-2^{\lfloor \log_2 x \rfloor}=1$（如果使用两位，是 01）。因而，$x=5$ 的 Elias-γ 编码通过合并 3 的一元编码（110）和 1 的两位二进制编码（01）得到，即 11001。另外一些 Elias-γ 编码的例子见表 9-2。不难看出，如果从左向右读，Elias-γ 码字可以唯一地解码。

Elias 介绍的另外一个编码方案是 Elias-δ 编码。Elias-δ 连接上面的那两个部分 1）和 2），但是 1）部分并不以一元码的形式表示，而是使用 Elias-γ 替代。对于 $x=5$，第一部分是 101，

而不是 110。所以，$x=5$ 的 Elias-δ 编码是 10101。表 9-2 展示了 Elias-δ 编码的其他例子。

通常来说，对任意一个整数 $x>0$，Elias-γ 编码需要 $1+2\lfloor \log_2 x \rfloor$位，而 Elias-δ 编码需要 $1+2\lfloor \log_2 \log_2 2x \rfloor + \lfloor \log_2 x \rfloor$位。对于较小的 x 值，Elias-γ 编码比 Elias-δ 编码更短；当 x 355
增大时，会产生相反的情况。因而，选择何种编码方式取决于我们要编码的值。

Golomb 表示另外一种对正整数 $x>0$ 编码的方法。这比较有趣，因为它能够通过调整参数来适应较小或较大的间距。对于某个参数 b，让 q 和 r 分别是 $x-1$ 除以 b 的商和余数（即 $q=\lfloor (x-1) \rfloor / b$ 和 $r=(x-1)-q \cdot b$）。然后，x 的编码就是：$q+1$ 的一元码表示后跟着 r 的二进制表示，使用$\lfloor \log_2 b \rfloor$或者$\lceil \log_2 b \rceil$位。如果 $r<2^{\lfloor \log_2 b \rfloor -1}$，那么 r 使用$\lfloor \log_2 b \rfloor$位（编码总是从一个 0 位开始）；否则，它使用$\lceil \log_2 b \rceil$位，其中第一位是 1，剩下的位对 $r-2^{\lfloor \log_2 b \rfloor -1}$ 的值在$\lfloor \log_2 b \rfloor$位内编码。例如，对于 $b=3$，有三种可能的余数，$r=0$、$r=1$、$r=2$，它们的编码分别是 0、10、11。类似地，对于 $b=5$，有五种可能的余数 r，从 0～4，它们的编码分别是 00、01、100、101 和 110。然后，如果对于 $b=3$，需要对 $x=9$ 编码，那么计算得到 $q=2$ 和 $r=2$。因此，编码就是 110 后面加上 11。对于 $b=5$，这些值变成 $q=1$ 和 $r=3$，最后的结果就是 10 后面加上 101。表 9-2 展示了 Golomb 编码对于 $b=3$ 的其他例子。

为了使用 Golomb 编码来对记录列表进行编码，必须为每个列表定义参数 b。一个合理的值是 $b \approx (N/l)\ln 2$，其中 N 是所有间距之和（或者列表中的最大数字），l 是列表中元素的个数。Golomb 编码通常比 Elias-δ 或 Elias-γ 压缩效果更好。例如，在 TREC-3 文档集中，使用文件寻址方案的 Golomb、Elias-δ 和 Elias-γ 编码，列表中每一项的编码的平均位数是 5.73、6.19 和 6.43 [1149]。这意味着相比于简单的倒排索引形式，在空间占用上减少到原来的 1/5（即压缩率大概为 20%）。一个使用我们之前介绍过的任一种间距编码技术压缩过的文件寻址的索引将占用 4%～8%的原始文本所需的空间。此外，如果我们用字节霍夫曼编码或密集编码（见 6.8 节）来压缩文本，那么整个压缩文本加上它的倒排索引将占用 30%～40%原始文本所需要的空间。

与 Elias 编码相比，Golomb 编码的缺点是需要遍历两遍文本，因为我们必须在压缩列表开始前就知道 N 和 l。因此，在索引构建时（见 9.2.5 节），我们不能直接以最终的压缩形式保存记录列表，但是可以用一些中间形式，如 Elias-δ 或 Elias-γ 编码。

如 9.2.3 节所见，间距编码的另一个复杂之处是，倒排表并不总是按顺序处理的，我们需要对某些操作进行随机访问，特别是在解决短语查询时。一个解决方法是，在列表中以规律的间隔保存数字的绝对值，通过只从它之前的采样点开始解压缩来加速对任意列表位置的访问。

现在我们考虑用于排序搜索的倒排索引。在这种情况下，在每个记录列表中的文档并没有按照文件标识符升序排列，而是按照索引项在文件中的频率，或者其他类似类型的权重降序排列。这使得压缩这些索引变得更加困难。然而，根据齐夫法则（见 6.5.2 节），可以认为许多词只在很多文件中出现一次，更一般地说，在每个记录列表中，会有很多频率一样的词。在这种情况下，在列表中出现频率相同的那些文件，仍然能够按照文件标识符以升序排列，这就使得能够使用间距编码来压缩列表的大部分。最近在 [56] 中显示，降低索引项在列表中出现次 356
数的精度可能会有好处，因为这会带来更好的压缩效果，而不会显著地降低检索质量。

9.2.7 结构化查询

在结构化文本上进行检索的算法，以及结构化信息保存的方式，很大程度上与每个检索模型相关（见第 7 章）。有些实现方法建立随机（ad hoc）索引来保存这个结构。这潜在地最大化了灵活性和效率，然而它需要额外的开发和维护成本。

描述任何特定的方法并不是我们的意图（如 XPath 和 XQuery 存在着许多实现方式），但是我们希望说明如何改造词寻址的全文倒排索引，使之能够回答某些有用的结构化查询。关于结构化文本的查询语言和搜索算法的更多细节将在第 13 章介绍。

让我们先假设结构在文本中用“标签”（即用于识别结构元素的字符串）来标记。这是 HTML 或 XML 中的情况，但并不是程序代码中的情况，在那里标签是隐性的，并从编程语言的语法中继承而来。索引算法的主要思想是像使用词语一样使用标签。在标签是隐性的情况下，倒排索引仍然能够用于那些实际上并未出现在文本中的假标签。在这步处理过后，倒排索引包含了回答结构化查询的所有信息。

例如，想象一个查询是“选择类型 **A** 的结构元素，它包括类型 **B** 的结构”。假设 **A** 类型（**B** 也类似）的结构被标签<A>和</A>包围，那么这个查询可以转换成找出<A>后跟着<B>，但两者之间没有</A>的情况。那些标签的位置可以通过全文索引得到，然后最后的答案就可以用类似于 9.2.3 节描述的解决短语查询的算法得到。不难看出，许多关于祖先、后代、祖先个数、后代个数、文本顺序等的查询都可以转换成先对标签进行搜索，然后验证记录序列。

在许多情况下，这个技术与随机索引方法一样有效，而且它与已有的文本数据库集成会更方便。

9.3 签名文件

签名文件（signature file）是基于散列的面向词的索引结构。它们有较低的开销（原始文本大小的 10%～20%），代价是必须顺序搜索索引。然而，尽管它们的搜索复杂度是线性的（而不是如之前的方法是次线性的），但它的常数是比较小的，这使得这项技术适用于并不十分大的文本集。尽管倒排索引在大多数应用中的效果超过签名文件，但签名文件在分布
357 式环境中仍会有一些应用，如高效地判断一个远程的索引是否包含对应给定查询的答案。

1. 结构

签名文件使用一个散列函数（或者叫“签名”）将词块映射成 B 位的位掩码。它把文本分成含 b 个词的块。对每个大小为 b 的文本块，分配一个长度为 B 的位掩码。这个掩码是通过对文本块中的所有词的签名进行按位或（OR）操作得到的。因此，签名文件不会比所有块的位掩码序列（加上指向每个块的指针）大。主要思想是，如果一个词出现在一个文本块中，那么在这个词的签名中置 1 的位在文本块的位掩码中也会置 1。因此当某位在查询词的掩码中置为 1，而在文本块的掩码中并未置 1 时，这个词就不在这个文本块中。图 9-10 展示了一个签名文件的例子，其中并未考虑文本中的禁用词。

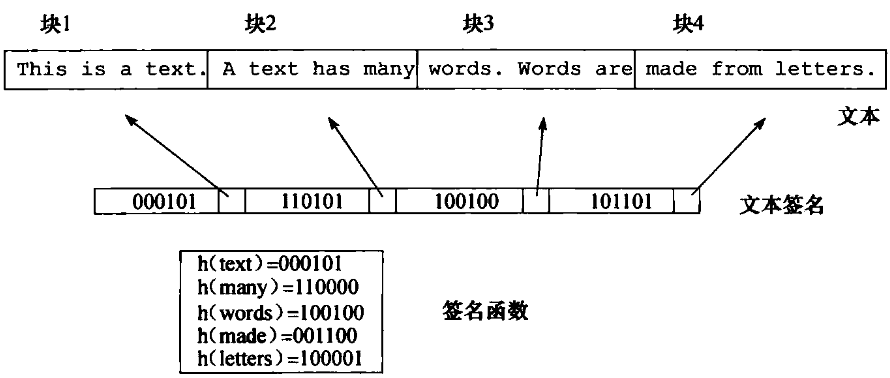


图 9-10 切成块的样例文本的签名文件

然而，有可能尽管词语并不存在，但对应的所有位还是都置 1 了。这叫做误检（false drop）。在签名文件的设计中，最精妙的部分在于：它保证误检的概率足够低，同时保证签名文件尽可能短。

散列函数要求产生的位掩码最少有 l 位置 1。较好的模型假设 l 位是在掩码中随机选择的（有可能重复）。令 $\alpha=l/B$。因为 b 个词中的每一个都随机选择 l 位置 1，所以掩码中某一位 1 在某个词的签名中也置 1 的概率是 $1-(1-1/B)^{bl}\approx 1-e^{-b\alpha}$ 。因而，在查询中 l 个置 1 的随机位在文本块的掩码中同样置 1 的概率是

$$(1-e^{-b\alpha})^{\alpha B}$$

它在 $\alpha=\ln(2)/b$ 时取最小值。在最优选择 $l=B\ln(2)/b$ 的情况下，误检的概率是 $(1/2^{\ln(2)})^{B/b}=1/2^{l}$ 。

因此，必须要决定合适的 B/b 比例。索引的空间代价大概是 $(1/80)\times(B/b)$，因为 B 的单位是位，而 b 是词的个数。然后，误检概率就是要花费的空间代价的函数。例如，10% 的代价表示误检概率接近 2%，而 20%的代价对应的错误概率大概是 0.046%。这里的错误概率对应于当检测一个匹配是否是误检时所需进行的顺序搜索的期望次数。 358

2. 搜索

搜索单个词的过程是将它散列成位掩码 W，然后将之与所有文本块的位掩码 B_i 比较。当 $W\&B_i=W$ 时（这里 & 是按位与操作），所有在 W 中置 1 的位在 B_i 中也置为 1，因此文本块可能包含这个词。因此，对于所有候选的文本块，必须执行一次在线遍历来验证词确实是在其中。不同于倒排索引，这个遍历无法避免（除非误检的风险是可接受的）。

在这种方案中，无法搜索其他类型的模式。另一方面，这个方案对于短语搜索和合适的邻近搜索效率更高。这是因为，所有的词都必须出现在块中，使得这个块能够包含短语查询或者邻近查询。因此，对所有查询掩码的按位或结果进行搜索，它们的所有置 1 位都必须出现。这减少了误检的概率。这是唯一在短语搜索中能提高性能的索引方案。

然而，需要注意检测块边界，以避免丢失跨块的短语。为了能够搜索 j 个词的短语或者 j 个词内邻近的短语，相邻的块必须重叠 $j-1$ 个词。

如果块对应于检索单元，那么通过限制相关的词都在一个块中，可以提高词或短语的简单布尔合取操作的性能。

我们只能找到来自 1992 年的真实的性能估计，它在一台带有本地磁盘的 Sun 3/50 机器上运行。在一个 2.8MB 的小型数据库上的查询时间为 0.42 秒。外推到当今的技术，我们发现性能大概接近每秒 20MB（它是线性时间的），因此在 250MB 的文本上大概需要 12 秒，这是十分慢的。

3. 构建

构建签名文件相对简单。将文本简单地切成块，给每个块生成一个签名文件条目。这个条目是对块中所有词的签名进行按位或操作得到的。

增加文本也很简单，因为只需要向签名文件中添加记录即可。文本删除通过删除合适的位掩码来实现。

除了把所有的位掩码按顺序保存外，还有其他的保存方案。例如，可以对掩码中的每一位用不同的文件保存，如一个文件保存所有的第一位，另一个文件保存所有的第二位……这减少了搜索查询的磁盘时间，因为只需遍历那些在查询中被置为 1 的 l 位所对应的文件。

4. 压缩

有许多替代方法可以压缩签名文件。所有这些方法都基于这样一个事实：在整个文件中

只有一些位是置1的。那么就有可能使用一些高效的方法对未置1的位进行编码，如游程编码（run-length encoding）。如果文件是以位掩码的序列形式保存，或者按每位一个文件的形式保存，那么就会有不同的考虑。这些压缩方法可以减少空间使用和硬盘访问时间，或者
359 在保持同样的空间代价时增加位掩码 B 的长度（以此降低误检的概率）。曾经有人报告其压缩率接近70%。

9.4 后缀树和后缀数组

倒排索引到目前为止仍然是推荐的信息检索系统实现方案。然而，它也有些局限性。文本必须能够方便地解析为词的序列，这样查询检索到的结果只能是整个词或者是由此组成的短序列。而且，不应该有太多不同的词，否则像词汇表这样的结构就会增长得过快，效率就会大幅度地下跌。

所有这些条件在许多语言（特别是西方语言）中都满足，但并不是全部。例如，芬兰语和德语是黏着语，这意味着短的语素（particle）连接起来才形成长的词，它相当于英语中的短语。通常我们不会检索那些长词，而是检索这些短的语素。自然语言解析工具可以用于分割这些语素，但是更简单、更健壮的选择是允许查询文本的任何子串，而不是定位到词语。

类似的问题也出现在一些东方语言中，如中文、日语和韩语。那些文本是在一个很大的字母表上的序列，在很多情况下每个符号都是表意文字。人类从符号的序列中区分出词语，因为他们理解文本，但自动分词仍然是一个悬而未决的研究问题。而且，对于那些语言，健壮的解决方案是让用户能够对符号流的任意子串都能搜索。

最后，有一些与信息检索无关的应用，但也存在子串搜索的问题，如计算生物学（DNA 或蛋白质序列）、音乐数据库（MIDI 序列）等。

在9.5节，我们将给出在文本中找出任何子串或者复杂模式的算法。然而，搜索时间随着数据库的大小按比例增长，因而这些方案并不适合很大的文本集。后缀树和后缀数组使得能够通过索引搜索匹配查询字符串或复杂模式的任何文本子串，同时搜索时间的增长速率低于文本集的增长。

这些索引将文本看做一个长字符串。文本中的每个位置当做一个文本后缀（即从那个文本位置到文本末尾的字符串）。例如，如果文本是 **"missing mississippi"**，那么后缀就是：

```
missing mississippi
issing mississippi
ssing mississippi
sing mississippi
ing mississippi
...
ppi
pi
i
```

不难看出，从不同位置开始的后缀是不同的，它们可以按字母进行比较（为此，我们在文本的最后加上了一个"$"字符，它比其他字符都小）。每个后缀被它的起始位置唯一地确定。

并不是所有的文本位置都需要索引。特别地，如果是在英文文本上只索引单词开头的情
360 况，那么产生的索引及其功能就与9.2.2节描述的全文倒排索引相当类似了。因为倒排索引在很多方面更方便，所以我们将关注在倒排索引不能使用的情况下如何使用后缀树和后缀数组。值得提到的是，后缀树和后缀数组在搜索长短语方面比倒排索引的效果更好，因为与搜

索词语相比，长短语并不需要什么特别对待。接下来我们假设所有的文本位置都被索引。

9.4.1 结构：trie 树和后缀树

1. trie

让我们首先简单地回顾一下什么是 trie 数据结构。trie 也叫做数字搜索树，是一个多叉树，它保存字符串集合，并能够在与字符串长度成正比的时间内检索任何字符串（与保存的字符串个数无关）。特殊字符“\$”被加到每个字符串的末尾，以此来保证没有任何一个字符串是另一个的前缀。在字符串集合 $\mathcal{P}=\{P_1,\cdots,P_r\}$ 上的 trie 树是一个识别 $P_1|\cdots|P_r$ 的树形 DFA（见 9.5.4 节）。因此，在$\mathcal{P}$中查找一个字符串相当于决定这个 DFA 是否能识别这个字符串。

在本质上，**后缀 trie 树**（suffix trie）是在文本 $T=t_1t_2\cdots t_n, t_n=$“\$”的所有后缀字符串上建立的 trie 数据结构。指向这些后缀 $t_i\cdots t_n$ 的指针保存在最终状态（即树的叶子）。为了减少 trie 树中的结点个数，后缀 trie 树移除了所有在叶子结点结束的无分支路径。图 9-11a展示了这个情况。例如，如果我们沿着从根到叶子标记为“**sing**”的路径走，那么我们看见路径在“**sin**”之后就被截断了。原因是在文本中只出现过一次“**sin**”，所以对应于后缀“**sing mississippi**”的无分支路径在那个点被截断了。但还是可以通过回到文本位置 4，并从那里开始读后缀来恢复整个后缀。

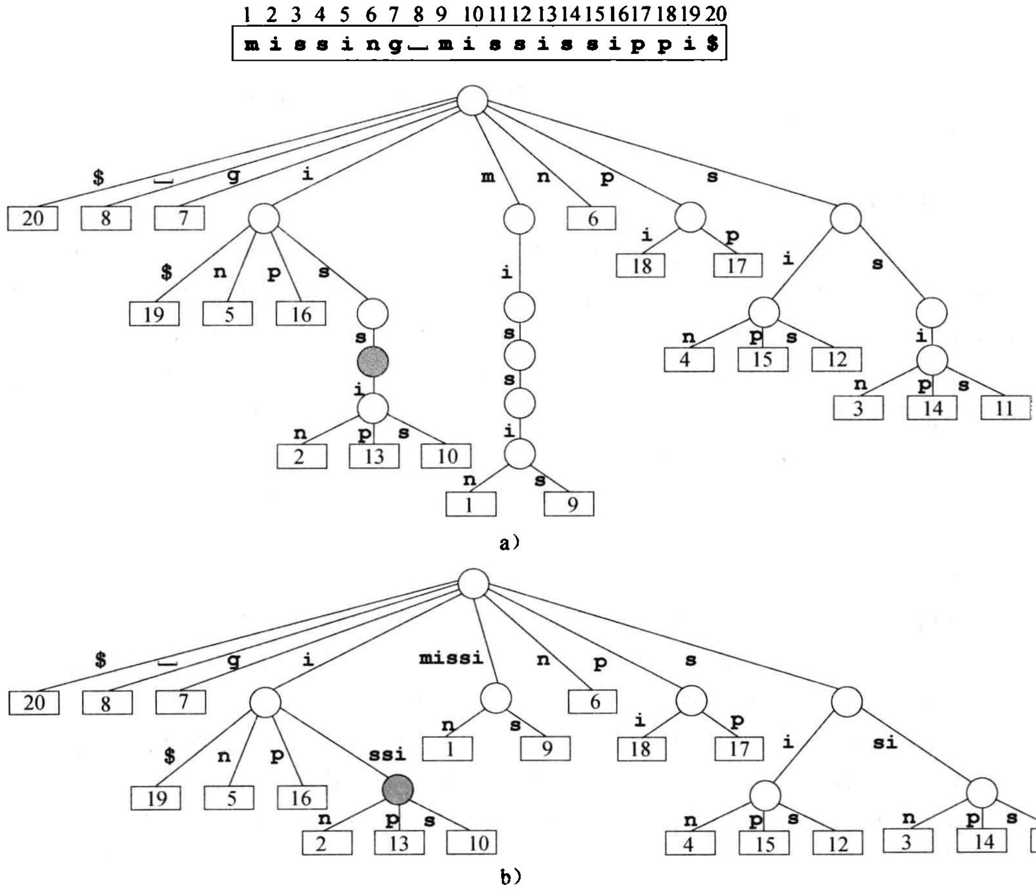


图 9-11 文本“missing mississippi”的后缀 trie 树 a）和后缀树 b）。两种情况下我们都用灰色显示了搜索“iss”时到达的结点

2. 后缀树

为了进一步减小所需要的空间，可以压缩所有剩余的无分支路径。压缩的结果就是*后缀树*（suffix tree）。因为有 n 个叶子，同时每个内部结点至少有两个子结点，所以后缀树的总大小是 $O(n)$ 。后缀树中的边一般是用字符串来标记。那些标签可以用指向字符串在文本中的记录指针和它的长度来表示，只占用常数级的空间，如图 9-11b 所示。

后缀树的问题就是它占用的空间。根据不同的实现，后缀树占用文本自身大小的 10～20 倍的空间。例如，1GB 文本的后缀树将需要至少 10GB 的空间。此外，后缀树在辅助存储器中表现得并不好，因此它只对相对较小的文本比较有吸引力。

后缀数组（suffix array）提供了与后缀树基本一样的功能，但是需要的空间要小得多。如果后缀树结点的子结点按照字典顺序从左向右排列边标签，那么后缀数组就通过从左向右的顺序收集所有叶子结点得到。更直接地，T 的后缀数组定义为一个指向 T 的所有后缀的数组，这里后缀已经按照字典顺序排列，如图 9-12 所示。

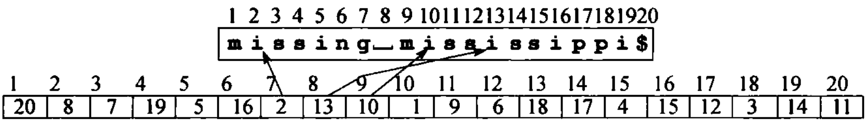


图 9-12 文本"**missing mississippi**"的后缀数组。箭头显示了模式"**iss**"的出现记录

后缀数组占用的空间通常是文本的 4 倍，这使得它对长文本更有吸引力（在 9.4.5 节，我们将展示后缀数组可以进一步压缩）。作为代价，后缀数组要比后缀树慢一点儿。在一些论文中，后缀树和后缀数组叫做 PAT 树和 PAT 数组。

9.4.2 简单字符串搜索

361~362 尽管在实现时，通常后缀树比后缀 trie 树更受推荐，但是大多数算法先在后缀 trie 树上解释更为简单。匹配给定模式 $P=p_1p_2\cdots p_m$ 的所有文本子串的主要性质是：*每个文本子串都是一个文本后缀的前缀*。因此我们在后缀 trie 树中寻找所有以 P 开头的后缀。

其主要思想是，按照 P 中的字符，在 trie 树中往下走。这使得我们到达所有以 P 开始的 trie 树后缀都需要经过的最后一个结点。从那个结点往下的叶子结点保存了以 P 开始的后缀的起始位置。也就是说，它们明确地指向 P 在 T 中的记录位置。

例如，以 $P=$"**iss**"和图 9-11 中的文本来说，我们用 P 中的连续字符驱动搜索过程，从 trie 树的根开始往下，并最终达到灰色的结点。然后我们考察以灰色结点为根的子树的叶子，并找到所有 P 在 T 中的记录：位置 2、13 和 10。

这个搜索过程有其他可能的输出：可能在 trie 树上没有路径能够拼写出 P，或者在我们读完 P 前就到达了叶子结点。在第一种情况下，我们知道 P 并未出现在 T 中（因为没有任何 T 的后缀以 P 开始）。在第二种情况下，假如 P'是当我们到达 trie 树叶子结点时已经读过的 P 的前缀。这意味着 P'在 T 中只出现过一次。P'的出现位置接下来的字符可能与 P 一样，也可能不一样。我们必须直接到叶子结点所指向的位置，验证 T 中是否出现 P。

如果搜索是在后缀树上进行的，那么问题会有另外的复杂性：因为边是用字符串标记的，而不是用单个字符。然而，对于从某个结点出发的标记边的所有字符串，它们的第一个字符是不一样的。因此，在任意一个结点，最多只能沿着一条边走。要想沿着某一条边走，边上所有的字符必须都匹配 P。也有可能在完全遍历完一条边之前，模式 P 就用完了。在这种情况下，答案就是以目标结点为根的整个子树。

例如，在图 9-11b 中，如果搜索“iss”，那么我们首先用第一个字符“i”来向下走。然后，我们应该尝试通过标记为“ssi”的边来向下走。这些字符匹配模式“iss”，但是它在读完整个边标签前就结束了。不管怎样，我们往下走到灰色结点，并找到正确的答案。

图 9-13 给出了后缀树搜索的伪代码。如果离开每个结点的边都被组织起来，使得能够在常数时间内通过这些边的第一个字符搜索到相应的边（例如，用一个由第一个字符索引的数组），那么后缀树能在 $O(m)$ 时间内返回带答案的子树。occ 个记录能在 $O(occ)$ 时间内获得。

```
Suffix-Tree-Search (S, P = p1 p2 ... pm)

  i ← 1
  while true do {
    if S is a leaf pointing to j then {
      if pi ... pm = t(j+i-1) ... t(j+m-1)
        then return S
        else return null
    }
    if there is an edge S --p'1...p's--> S' ∧ p'1 = pi then {
      j ← 0
      while j < s ∧ i + j ≤ m ∧ p'(j+1) = p(i+j) do j ← j + 1
      i ← i + j
      if i > m then return S'
      if j < s then return null
      S ← S'
    }
    else return null
```

图 9-13 后缀树搜索字符串 P 的伪代码。它返回以答案为根的子树，如果不存在的话则返回 null

搜索后缀数组略有不同。因为所有以 P 为前缀的后缀在词典顺序上是连续的，所以我们能够通过两次二分查找，找到以 P 为前缀的第一个和最后一个后缀，找到包含所有答案的后缀数组区间。图 9-12 展示了一个区间结果。注意，在这个二分搜索的每一步中都需要将 P 和一个文本后缀比较，因而搜索的代价是 $O(m\log n)$。

9.4.3 复杂模式的搜索

通常，用后缀 trie 树搜索复杂模式是通过模拟对应的序列算法和在 trie 树上回溯来实现的。例如，假设希望搜索某个正则表达式，我们如 9.5.4 节介绍的那样建立它的自动机，但并不加入自环（self-loop）。我们将检测到所有以匹配正则表达式的字符串开头的文本后缀。

因为这个原因，算法从 trie 树的根结点开始。对于当前结点的每个标签为字符 c 的子结点，向自动机输入 c，然后算法递归地进入子树。当递归从子树中返回时，恢复自动机在输入 c 363
前的状态。这个过程对当前结点的每个子结点重复进行。搜索以下面三种可能的形式停止：

1）自动机耗尽了所有的活跃状态。这意味着，从当前结点下来没有任何叶子结点能够匹配正则表达式。因此，我们抛弃当前分支，返回到父结点。

2）自动机到达结束状态。这意味着，从当前结点下来的所有叶子都是以匹配正则表达式的字符串开头的后缀。所以我们输出保存在叶子结点中的所有文本位置，并返回到父结点。

3）到达 trie 树的叶子结点。这意味着，我们必须在文本中继续验证，直到自动机到达了终结状态或者耗尽了所有的活跃状态。

已经证明，对于随机的文本，只需要遍历 $O(n^{\alpha}\text{polylog}(n))$ 个结点，其中 $0\leqslant\alpha\leqslant 1$，$\alpha$ 依

赖于正则表达式。这个搜索时间对于大多数感兴趣的正则表达式是次线性的（如果 $\alpha<1$）。扩展的模式也可以通过把它们当做正则表达式用同样的方式来搜索。

带索引的容错度为 k 的近似字符串匹配（见 9.5.6 节）也可以用同样的思想实现。我们使用同样的回溯方法，但在运行算法时，计算匹配 trie 前缀和 P 所产生的错误个数。为此，
364 必须对式（9-2）中 C 的循环进行改变，使初始化第一行时 $C_{0,j}=j$。当从字符 c 向下走，就会根据 c 计算矩阵 C 的一个新列。当列中所有的值都超过 k 时，就抛弃一个分支；如果最后一列的最后一个单元的值并未超过 k，那么输出所有的叶子结点。对于基于自动机的近似搜索，这个过程与基于自动机的正则表达式搜索一致。

在 trie 树上的近似搜索深度不会超过 $m+k$，因此，对于足够短的模式，时间复杂度独立于文本大小（因为我们不可能访问超过 $O(\sigma^{m+k})$ 个结点）。对于长模式，在搜索时间上依赖 m 的指数复杂度就越来越明显，这样就必须使用其他方法了。一个高效的技术就是将 P 分割成 j 分片，对每个分片的搜索允许 $\lfloor k/j \rfloor$ 个错误，然后直接在文本中验证分片的邻域，从而得到整个 P 的记录。这是对序列搜索技术的扩展。如果合理选择 j，可以得到平均的搜索时间是 $O(n^{\alpha}\text{poly}(m))$，其中对于足够小的 k/m，有 $\alpha<1$。

后缀树能够执行我们还未考虑过的其他复杂搜索。这些特别的操作在一些特别的应用中有用。有些例子是：找出在文本中出现多于一次的最长子串（即最深的后缀 trie 树结点），找出固定长度的最常见子串等。

我们所介绍的用于后缀 trie 树的算法显然可以用于后缀树。它们同样也适用于后缀数组，不过增加了一个 $O(\log n)$ 的时间惩罚因子。每个后缀 trie 树结点对应于一个后缀数组区间，这个区间包含以此结点为根的子树的所有叶子。如果从 trie 树的根到某结点的路径拼写出了字符串 S，那么对应的后缀数组区间就是那些以 S 开始的后缀。根结点对应于整个后缀数组，叶子对应于单个数组单元。我们通过维护对应于当前后缀 trie 树结点的数组区间来模拟后缀数组遍历。在根据字符 c 在后缀 trie 树中从一个结点往下走的时候，我们二分查找当前的后缀数组区间所对应的子区间。

9.4.4 构建

一段包含 n 个字符的文本可以在 $O(n)$ 时间建立其后缀树。然而，如果内存无法放下后缀树，那么这个算法的性能将很差；当后缀树需要大量空间时更是如此。这个算法的参考文献将在本章的最后给出。

我们关注直接构建后缀数组。因为后缀数组是以字典序排列的文本指针的集合，所以生成它的一个简单方法就是，用任何经典的排序方法，按字典序排列所有指针所指向的后缀。注意，为了在这个排序中比较两个后缀数组条目，必须访问对应的文本位置。这些位置基本上是随机的。所以，即使使用外存排序算法，至少文本应该保存在内存中，以便得到可接受的性能。一个好的排序算法需要 $O(n\log n)$ 次字符串比较，以及平均 $O(n\log^2 n)$ 的时间。

不幸的是，如果文本包含很长的重复子串，那么这个简单的构建方法就会彻底失败，因为在某些时候，将比较两个从重复串开始的后缀，而这些比较需要检验很多字符才能确定字
365 典顺序。

有一些为后缀数组特别设计的更强大的排序算法，在本章最后将给出一些参考文献。大多数算法的主要想法是：如果知道 $t_{i+1}\cdots t_n<t_{j+1}\cdots t_n$ 和 $t_i=t_j$，那么我们不用比较那些后缀就可以推断出 $t_i\cdots t_n<t_j\cdots t_n$。例如，假设所有的后缀已经按照它们前 2^{l-1} 个字符排序。那么，不需要任何字符串比较就可以通过它们的前 2^l 个字符对它们排序。假设我们在一个后

缀数组区域中，其中所有后缀的前 2^{l-1} 个字符是相同的，我们想要完善它，使之形成一些所有后缀的前 2^l 个字符都相同的桶。为了确定 $t_i \cdots t_{i+2^l-1}$ 是比 $t_j \cdots t_{j+2^l-1}$ 小、相等还是大，我们只需要找出（如用反向排列）$t_{i+2^{l-1}} \cdots t_{i+2^l-1}$ 的桶是在 $t_{j+2^{l-1}} \cdots t_{j+2^l-1}$ 的桶之前、一样还是之后。基于这个思想，以不同的方式创建了不同的算法。现在已经有十分快速的算法，允许很长的重复子串，并且只需要很少的额外空间用于构建。

对大文本构建后缀数组

上面的算法假设文本能够放入内存中。当这个条件不满足时，就需要一个针对外存的特定算法。我们展示一个在实际中表现很好的算法。

这个算法把文本分割成能够在内存中排序的块。然后，对于每个块，它在内存中建立块的后缀数组，并与前面的文本块建立的后缀数组进行合并。即

1）为块 1 建立后缀数组。
2）为块 2 建立后缀数组。
3）合并块 1 和块 2 的后缀数组。
4）为块 3 建立后缀数组。
5）合并块 3 和块 1＋2 的后缀数组。
6）为块 4 建立后缀数组。
7）合并块 4 和块 1＋2＋3 的后缀数组。
8）……

困难的部分在于，如何把对应于块 1、2、…、$i-1$ 的较大的后缀数组 LA（已经建立）与对应于块 i 的较小的后缀数组 SA（刚建立）合并。朴素的合并方法还是需要比较文本位置，这些位置遍布在大量文本中，所以这个内存问题还是存在。解决方案就是先确定有多少个 LA 的元素将放到 SA 中两个连续元素之间，然后用这些信息来合并数组，而不用访问文本。这些信息保存在一个计数数组 C 中，$C[j]$ 表示有多少个 LA 的后缀按字典序排在 $SA[j]$ 和 $SA[j+1]$ 之间。一旦 C 计算出来了，LA 和 SA 只用一次顺序扫描就可以方便地合并起来，扫描期间把数组内容追加到初始为空的输出中：

1）追加 LA 中前 $C[0]$ 个元素。
2）追加 $SA[1]$。
3）追加 LA 中接下来的 $C[1]$ 个元素。
4）追加 $SA[2]$。
5）追加 LA 中接下来的 $C[2]$ 个元素。 366
6）追加 $SA[3]$。
7）……

剩下的关键点就是如何计算计数数组 C。这不用访问 LA 就能得到。确切地说，是将对应 LA 的文本（即 T 的前 $i-1$ 块）顺序读入内存中。并在 SA 中搜索文本的每个后缀（在内存中）。一旦确定文本后缀按字典序位于 $SA[j]$ 和 $SA[j+1]$ 之间，我们对 $C[j]$ 的值加 1。图 9-14 展示了对一个块的整个处理过程。

这个算法的 I/O 代价是 $O(n^2/M)$，这里的 M 是进程中可用的内存数量。CPU 的代价是 $O(n^2\log(M)/M)$。这两个代价远不是最优的，但是这个算法在大多数实际情况中表现得都非常好。

注意，同样的算法可以用于索引维护。如果将一个大小为 n' 的新文本加入数据库中，那么它能像之前一样分成块，然后按块合并到当前的后缀数组中。这将花费 $O(nn'\log(M)/M)$ 的

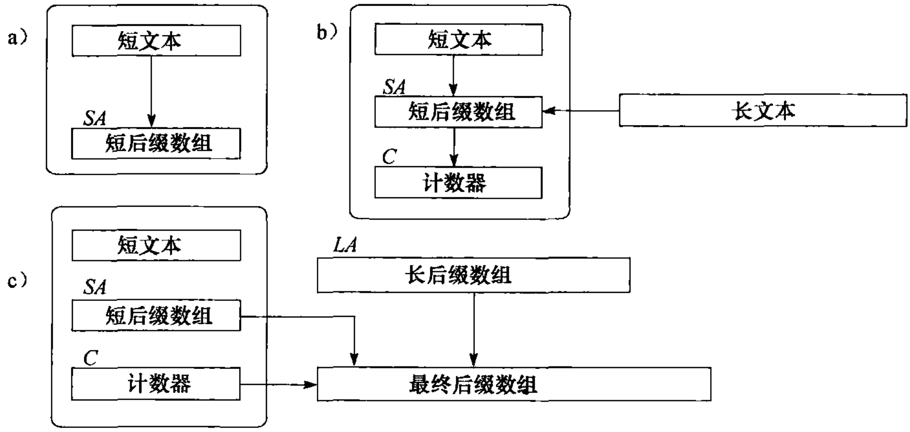


图 9-14 对大文本构建后缀数组的步骤：a）建立局部后缀数组 SA；b）计算计数数组 C；c）合并后缀数组 SA 和 LA

CPU 时间。为了删除某些文本，需要进行一次 $O(n)$ 时间的数组遍历，删除所有在需要删除的区域中的文本位置，然后把剩下的位置压缩成一个连续的数组。

9.4.5 压缩的后缀数组

后缀数组压缩的情况更加独特，因为我们将研究压缩后缀数组替代文本，原因是它们能够重新产生文本子串，因此并不需要保存文本。这就叫做*自索引*（self-indexes）。

正如 9.4 节提到的，后缀数组的一个重要问题就是它们的空间需求很大。直到数年之前，后缀数组一直认为是基本上随机排列的，因而不可压缩。但这个情况从 2000 年开始大大地改变了，当时几个几乎同时出现的研究成果表明，后缀数组实际上是可以高度压缩的。

367

再次看图 9-12 中的后缀数组，称之为 $A[1,n]$。$A[15..17]$ 的值是 4、15、12。所有值减 1，同样的序列在 $A[18..20]$ 也发现了；再减 1，在 $A[7..9]$ 又发现了。通过观察图 9-11，不难看出这个规律的原因：区间 $A[15..17]$ 对应于通过字符串“**si**”所到达的后缀树的子树。因为文本中所有的“**si**”都以“**s**”为前导，所以“**ssi**”的子树包括了“**si**”的位置平移 1 的位置。对于“**issi**”也是同样的道理。

如果压缩器从后往前读取文本，那么它能够知道，每当读到“**si**”（反序）时，接下来它看到的字符将是“**s**”，然后是“**i**”。这与 6.8.3 节介绍的 k 阶压缩相关。已证实，那些能被高阶压缩器压缩的文本，其后缀数组会有更多像上面所说的那种规律。通过利用这些规律，可压缩文本的后缀数组也能被压缩。

我们接下来说明后缀数组压缩的两种主要方法。对用那些使用压缩技术的后缀数组进行操作实际上比不使用压缩的后缀数组要慢。然而，当文本太大，以至于它的后缀数组无法放到内存中时，压缩的形式可能是相当合算的，因为它避免借助磁盘来搜索文档集。与压缩的倒排索引的例子相反，在磁盘上使用后缀数组索引（压缩或未压缩的）并没有很多的发展。

1. 使用 Ψ 函数

揭示上文提到的后缀数组规律的一个方法是通过 Ψ 函数，其定义如下

$$A[\Psi(i)] = A[i] + 1$$

其中，当 $A[i] = n$ 时，$A[\Psi(i)] = 1$。也就是说，$\Psi(i)$ 表示 A 中当前元素的值加 1 所对应

的元素在数组中的位置。图 9-15 说明了 Ψ 函数的计算，以及 Ψ(*i*) − Ψ(*i* − 1) 的值。

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	m	i	s	s	i	n	g	␣	m	i	s	s	i	s	s	i	p	p	i	$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	20	8	7	19	5	16	2	13	10	1	9	6	18	17	4	15	12	3	14	11
Ψ	10	11	2	1	12	14	18	19	20	7	9	3	4	13	5	6	8	15	16	17
diff	10	1	−9	−1	11	2	4	1	1	−13	2	−6	1	9	−8	1	2	7	1	1
	$	␣	g	i						m		n	p		s					

图 9-15　对样例后缀数组计算函数 Ψ，同时 $diff(i) = \Psi(i) - \Psi(i-1)$ ($diff(1) = \Psi(1)$)。我们标出了数组中开头字符相同的后缀组成的区域

Ψ 的一些属性在图中比较明显，也不难证明。第一，在后缀以同样的字符开始的区域中，Ψ 的值是递增的，因此它可以分割成 σ 个递增的区间。然后就可以用它的第一个绝对值 368
和在相邻数字间的（正）间距来代表每个区间。如果对这些间距使用 Elias-δ 编码（见 9.2.6 节），那么 Ψ 就可以粗略地压缩成文本的零阶熵（把它看做字符的序列，见 6.8.3 节）。第二，在我们之前指出的规律出现的区域（这个例子中是 *A*[18..20] 相对于 *A*[15..17]，*A*[7..9]相对于 *A*[18..20]），它满足条件：Ψ(*i*) − Ψ(*i* − 1) = 1 。这使得能够对 Ψ 的差分编码中的多个 1 使用游程压缩算法。把这两种技术结合起来能极大地压缩 Ψ。快速访问 Ψ 的任意位置可以通过以固定的采样间隔保存 Ψ 的绝对值。

更为有趣的是，*T* 中的字符不用访问 *T* 就能得到。因此，*T* 实际上可以删除，它的任何子串都可以只通过 Ψ 和一些小的辅助结构得到。假设希望不用 *A* 或 *T* 就知道从 *A*[*i*] 开始的后缀的第一个字符是什么，即 $t_{A[i]}$ 。而我们已经能够知道 *T* 中每个字符所对应的区间（见图 9-15 的底部；这实际上对应 6.8.6 节的表 *C*），并能确定包含 *i* 的区间。例如，$t_{A[7]}$ 的第一个字符肯定是 "i"，因为 7 ∈ [4,9]，[4, 9] 是以 "i" 开始的后缀的区间。为了知道 *A*[*i*] 所指向的后缀的第二个字符，即 $t_{A[i]+1}$ ，我们计算 *i*′ = Ψ(*i*) ，并使用上面的方法找出 $t_{A[i']} = t_{A[i]+1}$ 。后续的字符可以通过重复计算 Ψ 来一个个地得到。例如，为了找到字符 $t_{A[7]}\cdots$ ，我们已经得到 $t_{A[7]}$ = "1"。现在 i′ = Ψ(7) = 18 。因为 18 在字符 "s" 的区间里，所以我们得到 $t_{A[7]+1} = t_{A[18]}$ = "s"。为了得到第三个字符，我们计算 i″ = Ψ(18) = 15 ，它同样在 "s" 的区间中。因此，后缀 $t_{A[7]}\cdots$ 以 "iss" 开始。注意，我们得到这些的过程中并没有知道 *A*[7] 和访问 *T* 或 *A*。

这样的机制使得我们能够二分查找 *A*，而不必拥有 *A* 或者 *T*。当一个后缀需要与查询比较时，我们使用上面所描述的机制来得到后缀的后续字符。因此，如果我们能够在常数时间内访问 Ψ，那么搜索的时间复杂度是 $O(m\log n)$ ，和未压缩形式相同。最后，我们得到，答案在区间 *A*[*sp*,*ep*] 中，可以立即知道出现次数是 *ep* − *sp* + 1 。

然而，这常常是不够的，因为通常希望知道模式 *P* 出现的文本位置，而不是在 *A* 中的区间。我们仍然不需要得到 *A*，以便对 *sp* ≤ *i* ≤ *ep* ，显示出记录的位置 *A*[*i*] 。

为了能够在 *T* 中定位记录，我们以 *T* 的规律间隔对 *A* 采样：在 *T* 中每隔 *s* 个字符，记录指向该文本位置的后缀数组位置。即，对于每个 1 + *j* · *s* 形式的文本位置，使得 *A*[*i*] = 1 + *j* · *s* ，然后我们将 (*i*,*A*[*i*]) 对保存在按第一个分量可搜索的词典中。如果在搜索时需要显示后缀数组元素 *A*[*i*] 的内容，而 (*i*, ∗) 并不在采样的子集中，那么我们尝试 *i*′ = Ψ(*i*) ，

然后是 $i''=\Psi(\Psi(i))$ 等，直到找到 A 的一个采样过的单元。假如，在 k 次调用 Ψ 函数以后，我们找到一个采样过的单元 $(\Psi^k(i),A[\Psi^k(i)])$。然后，因为 $A[\Psi^k(i)]=A[i]+k$，所以原始查询的答案就是 $A[i]=A[\Psi^k(i)]-k$。同样需要注意，因为我们采用固定的间隔采样 T，所以我们对于那些 $k\leqslant s$ 的情况，肯定能找到一个采样过的元素。

最后，我们应该也能够展示任何文本子串，因为我们计划舍弃 T。注意，我们已经知道
369 对于给定 i，如何获得 T 中从文本位置 $A[i]$ 开始的字符。为了显示 $T[l,r]$，我们再次采用采样机制：查看在 T 中 l 之前的最近一次采样的位置，并显示从那里开始的文本。为此，我们对采样 $(i,A[i])$ 按照在数组中的文本位置排序，则最近采样的 i 分量是在那个数组的 $\lceil l/s\rceil$ 位置上。

因此，最终的压缩后缀数组就可以通过压缩的数组 Ψ（带有绝对值采样以快速直接访问）、表 C 和采样 $(i,A[i])$ 以及它的访问结构来形成。整个结构所需要的空间依赖于文本的类型和采样的密度。一般来说，使用 T 所需要的 30%～70%的存储空间就能得到可观的性能。我们强调，这个索引不仅替代了后缀数组索引 A，而且还替代了文本 T 本身。所以这个方案比原始的文本占用更少的空间，同时提供了访问 T 的索引方式。

2. 使用 Burrows-Wheeler 变换

另一个完全不同的压缩和搜索后缀数组的方法是通过 6.8.6 节所介绍的 Burrows-Wheeler 变换（Burrows-Wheeler transform，BWT）。T 的 BWT 可以通过连接 A 中每个后缀之前的字符，即 $t_{A[i]-1}$（如果 $A[i]=1$ 则为 t_n）得到。例如，$T=$ "**missing Mississippi $**" 的 BWT 是 $T^{bwt}=$ "**ignpssmsm $ ipisssiii**"。已经证实，BWT 倾向于把相同的字符合并成连续的一段区间，而且有一些大的区域，其中不同字符很少。事实上，不难把 BWT 的连续区间和在函数 Ψ 中出现的情况联系起来。在 6.8.6 节，我们已经讨论过 BWT 的可压缩性质。现在我们只关注如何只使用 T 的 BWT 在 A 中搜索（即我们同样不使用 A 和 T）。

方案就是对 6.8.6 节的 LF 映射的泛化。我们使用同样的数组 C，以及一个扩展函数 $Occ(c,i)$，它给出了字符 c 在 $T^{bwt}[1,i]$ 中出现的次数。我们以反向的形式搜索 $P=p_1p_2\cdots p_m$：首先在 A 中找到其中的所有后缀都以 p_m 开始的区间，然后找到以 $p_{m-1}p_m$ 开始的区间，然后 $p_{m-2}p_{m-1}p_m$ 等，直到我们找到了 P 的答案。

第一步是简单的：p_m 的记录在 $A[C[p_m]+1..C[p_m+1]]$。然后，假设我们已经知道了 $p_i\cdots p_m$ 在 A 中的区间 $A[sp_i..ep_i]$，那么 $p_{i-1}p_i\cdots p_m$ 在区间 $A[sp_{i-1}..ep_{i-1}]$，其中 $sp_{i-1}=C[p_{i-1}]+Occ(p_{i-1},sp_i-1)+1$，以及 $ep_{i-1}=C[p_{i-1}]+Occ(p_{i-1},ep_i)$。

例如，为了搜索 "**iss**"，我们从 $(sp_3,ep_3)=(15,20)$ 开始，因为 $C[\text{'s'}]=14$ 和 $C[\text{'t'}]=20$。然后，"**ss**" 的区间 $(sp_2,ep_2)=(14+Occ(\text{'s'},14)+1,14+Occ(\text{'s'},20))=(14+3+1,14+6)=(18,20)$（参看图 9-15，$A[18..20]$ 是指向 T 中的 "**ss**" 记录的区域）。最后，我们计算出 $(sp_1,ep_1)=(7,9)$，即得到最终的答案 $A[7..9]$。

为了定位记录，我们使用了一个与 Ψ 函数类似的采样机制。不同的是，我们使用 LF 来进行反向遍历，而不是使用 Ψ 来正向地遍历文本。注意，为了计算 LF，我们需要访问 T^{bwt} 的任意位置。

这个方法中的挑战，就是在 T^{bwt} 上高效地计算 $Occ(c,i)$，而不需要使用很多空间。如
370 果 T^{bwt} 没有保存，那么我们还必须能够对于任意 i 都能访问 $T^{bwt}[i]$。我们展示如何使用一个叫小波树（wavelet tree）的数据结构在 $O(\log\sigma)$ 时间内在 T^{bwt} 上进行这两个操作。

字符串 S 的小波树是一个平衡二叉树，其中每个结点处理字母表符号的一个子集。根结点处理整个字母表，而叶子结点处理单个符号。每个结点处理的符号的集合被分成两个子

集，这样每个子结点就处理一半。对于每个结点，我们给它指派 S 的一个子序列，其中的符号都属于该子集。这个子序列并未真正地保存。每个结点保存一个位图，对于子序列中的每个符号，位图表明了该符号应该去它的左子结点还是右子结点。很容易看出，小波树的高度是 $\lceil \log\sigma \rceil$。

图 9-16 展示了 $S = T^{bwt}$ = **“ignpssmsm$ ipisssiii”** 的小波树。假如我们希望知道在位置 13 的符号。因为根结点位图中的位置 13 上是 1，所以我们知道这个字符是在右分支上，所以我们移到根结点的右子结点上。而且，因为位置 13 包含的是第 8 个 1，所以现在我们所考虑的位置是 8，而不是根结点上的 13 了。根结点的右子结点的第 8 位还是 1，而且是第 5 个 1，所以我们再次向右走，并查看位置 5。那个位置上是一个 0，而且是第 2 个 0。所以我们向左走，并考虑位置 2。到目前，我们已经到达了 **“p”** 叶子，因此 $S[13]$ = “p”。

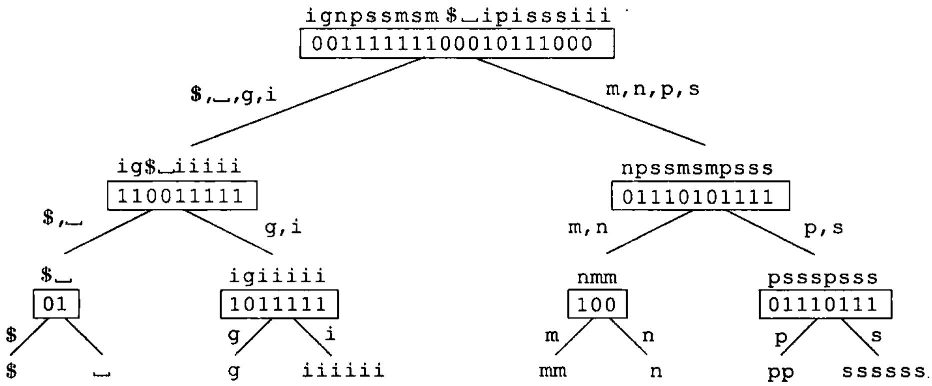


图 9-16 序列 **“ignpssmsm$ipisssiii”** 的小波树，只有位图是真正保存的

现在让我们看看如何使用小波树在 T^{bwt} 上计算 $Occ(c,i)$。假设我们希望计算 Occ(‘i’, 13)。我们知道 **“i”** 属于根结点的左儿子，因此在根结点位图上它的记录就标记为 0。到位置 13，总共有 5 个 0 位，所以我们移到左子结点，并考虑直到位置 5 的情况。现在，我们知道 **“i”** 属于右子结点，它在这个位图中标记为 1。到位置 5 有 3 个 1，所以我们去右子结点，并考虑直到位置 3 的情况。再次，我们知道 **“i”** 属于右子结点，所以我们计算到位置 3 为止 1 的个数，并发现有 2 个 1。我们到达叶子 **“i”** 的位置 2。显然，在 **“iiiii”** 中，到位置 2 为止总共有 2 个 **“i”**。所以最后的答案就是 Occ(‘i’,13) = 2 。

注意，在位图上我们只需要一个操作：计算到给定位置 i 为止 1 的个数（或者是 0 的个数）。这个操作称为 $rank(i)$，而且它能高效地完成，并使用很少的空间：我们每隔 s 个位置保存 $rank(i)$ 的答案，然后对于任何 $rank(i)$ 查询，从 i 之前最近采样的位置开始“暴力”计算。 371

未进行任何压缩，小波树基本上需要 $n\log\sigma$ 的空间，也就是说，与序列 S（或 T）一样。通过变成 T 的霍夫曼树的形状（而不是平衡树，见 6.8.4 节），需要的空间就变成与 T 的零阶熵（以字符计）成线性关系。搜索的复杂度（不计算记录定位）可以达到 $O(m\log\sigma)$。

因此，基于 BWT 的索引的简单实现基本地由 T^{bwt} 的小波树、数组 C 和使用 Ψ 时类似的一些采样对 $(i,A[i])$ 等组成。更复杂的实现见本章最后的参考文献部分。取得的压缩性能与基于 Ψ 的方法类似，但是基于 BWT 的方法通常在 DNA 等小字母表上表现得更好，而基于 Ψ 的方法在英语文本等较大的字母表上表现得更好。

9.5 序列搜索

如 9.2.2 节所示，全文检索的倒排索引在某些情况下会采用序列搜索：扫描词汇表，在

文件或块寻址时找到短语，在块寻址时完成任何搜索，在文件寻址时在界面中加亮显示记录的情况等。本节将对不同类型的模式介绍序列搜索算法。

通常，序列搜索问题是：给定一段文本 $T=t_1t_2\cdots t_n$ 和一个表示字符串集合的模式 P，找出 P 中字符串在 T 中的所有记录。而且在 T 上未建立任何结构。模式匹配的领域十分广泛，所以本节的意图只是展示那些最实用的算法，它们用于解决信息检索场景中可能会出现的最常见的问题。我们在本章的最后给出了进一步阅读的一些参考文献。

我们从最简单的情况开始：模式只表示单个字符串 $P=p_1p_2\cdots p_n$。这叫做*严格串匹配*(exact string matching)。通过一些很小的变化，这个问题包含 7.2.1 节介绍的许多基本查询，如词、前缀、后缀和子串搜索。我们从一个作为基准的“暴力”方案开始，然后展示 Horspool 算法。这个算法实现起来很简单，而且在大多数情况下是最快的。我们还介绍了一个变种方案，当模式更长或者词汇表更短时，它的效果很好（见第 6 章）。

然后，我们介绍稍微复杂一些的模式，其中仍然只有一个模式框架 $p_1p_2\cdots p_m$，但是每个位置表示字符的集合。这表示一些类型的通配符或大小写不敏感的搜索。此外，我们让某些位置 p_i 是可选的，或者是可重复的。所有那些问题都在一个简单、通用的基于非确定自动机和位并行的框架中解决。

（有限）自动机所能搜索的、最复杂的模式是正则表达式。下面，我们将简短地介绍基于自动机的、搜索正则表达式的方法。

接下来，我们讨论多模式搜索。多模式搜索可能会因为如下情况而出现，如查询扩展或者词干提取过程，以及批量处理很多查询等。此外，当在一个块寻址的索引（见 9.2.2 节）或者使用基于词的模型压缩的文本（见 6.8 节）上搜索复杂模式时，多模式匹配看起来是自
372 然而然的。虽然一般情况下只会搜索少量字符串，但在某些情况下，许多字符串是同时搜索的。我们介绍扩展 Horspool 算法和基于自动机的算法来处理多模式搜索。

最后，我们考虑近似搜索。它们在文本中找出足够接近搜索模式的字符串所在的记录。近似搜索能够从输入、拼写、光学字符识别和其他可能出现在文本或模式中的错误中恢复过来。

我们假设字符串是从大小为 σ 的字母表 Σ 中抽取的字符的序列（如使用 ASCII 码，Σ 可以是 [0，255]），并从位置 1 开始。在平均情况分析中，我们假设文本中的每个字符都是均匀、独立地从 Σ 中选择的。在 9.5.7 节，我们将看到本节所开发技术的进一步应用。

9.5.1 简单字符串：Horspool

Horspool 算法处于一个十分幸运的位置：十分容易理解和编程，同时在很多情况下又是最快的，特别是当搜索自然语言文本时。

当了解了所谓的*暴力*（brute force）算法后，就更容易理解 Horspool 算法。暴力算法就是简单地试遍在文本中所有可能的模式位置，并一个接一个进行检查。更明确地说，这个算法在文本上滑动一个长度为 m 的窗口 $t_{i+1}t_{i+2}\cdots t_{i+m}$，其中 $0\leqslant i\leqslant n-m$。每个窗口表示模式的一个潜在记录，必须验证它是否与 $P=p_1p_2\cdots p_m$ 相等，并加以汇报。一旦验证之后，这个算法就滑动窗口到下一个位置。

图 9-17a 展示了在文本 $T=$ “**abracabracadabra**”上搜索 $P=$ “**abracadabra**”。第一个文本窗口是“**abracabraca**”。当验证它并不匹配 P（在模式字符“**d**”之后失败）之后，这个窗口平移一个位置。现在的文本窗口是“**bracabracad**”，这里验证在第一次尝试时就失败了；然后我们再次平移一个位置，以此类推。暴力算法在最坏的情况下要花费 $O(mn)$ 时间，在

平均或最好的情况下是 $O(n)$ 。

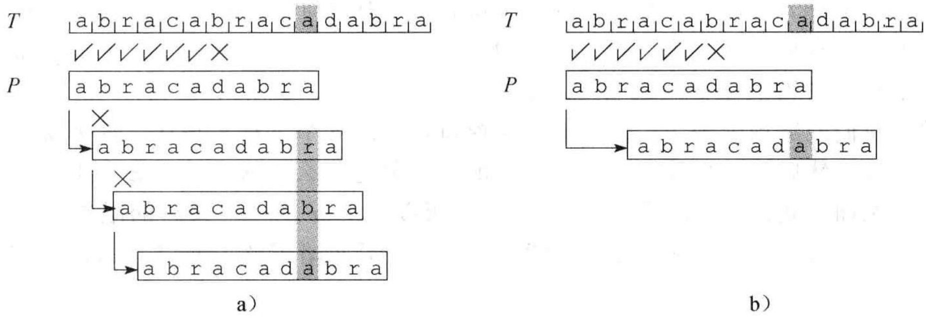


图 9-17 一段简单的文本及模式：a）使用"暴力"方法和 b）Horspool 算法搜索

这个图还用阴影显示了与第一个滑动窗口中最后一个字符对齐的那些模式字符。因为它们在前两次平移中并不匹配，所以那些验证必然会失败（然而暴力算法在不同的模式位置上发现了不匹配）。关键点就在于，我们可以通过合理地预处理这个模式来预测开始的两次平 373
移是无用的。

Horspool 算法的想法是，以更聪明的方式平移窗口，以此来跳过一些位置，而不丢失模式的任何记录。该算法预先计算一个用字母表中的字符索引的表 d：$d[c]$ 表示，如果窗口的最后一个字符是 c，那么窗口可以平移多少个位置。换句话说，$d[c]$ 是从模式的末尾到模式中最后出现 c 的位置（除 p_m 之外）的距离。

图 9-17b 用 Horspool 算法重复了这个例子。在验证了窗口并不匹配 P 后，我们检验窗口的最后一个字符"a"。因为 $d[\text{'a'}]=3$ ，所以这个窗口就有把握平移 3 个位置，这样 P 中的第 4 个"a"就与窗口最后的"a"对齐了。

如图 9-18 的伪代码所示，Horspool 算法极其容易编写。Horspool 算法在最坏的情况和暴力算法一样，但在最好的情况下只需要 $O(n/m)$ 时间。如果字母表与 m 相比足够大，这也是在平均情况下的复杂度。大多数自然语言搜索满足这种情况，因此 Horspool 算法通常是最好的选择。

Sunday 对 Horspool 算法做了一个小的改进，他使用在文本中的下一个字符来做下一次平移（即 p_{j+1}），因为我们在当前位置已经舍弃或者找到了一个记录。这对于小模式提高了跳跃的长度，但是并未减少搜索时间，因为在最好情况下，我们需要查看两个字符而不是一个。

```
Horspool (T = t1t2...tn, P = p1p2...pm)

for c ∈ Σ do d[c] ← m
for j ← 1...m − 1 do d[pj] ← m − j
i ← 0
while i ≤ n − m do {
  j ← 1
  while j ≤ m ∧ t(i+j) = pj do j ← j + 1
  if j > m then report an occurrence at text position i + 1
  i ← i + d[t(i+m)]
}
```

图 9-18 Horspool 字符串匹配算法的伪代码

还有许多其他字符串匹配算法，这里并未涉及，因为它们在实际中慢得多。这些算法包括第一个线性算法（Knuth、Morris 和 Pratt[912]），以及第一个提出 Horspool 算法中所使用的启发式方法的 Boyer-Moore 算法［246］。这些算法在实际文本中的比较见图 9-38。 374

小字母表和长模式

当在小字母表上搜索长模式时，Horspool 算法表现得并不好，因为它的平均搜索时间相比 $O(n/m)$ ，更接近 $O(n/\sigma)$ 。例如，想象一下，一个计算生物学的应用，需要搜索在字母表 {A，C，G，T} 基础上的包含 300 个核苷酸的字符串：此时平均平移长度只有 4。

通过在平移窗口时考虑连续的字符对，小字母表的问题可以得到缓解。正如我们将模式与最后的窗口字符 t_{i+m} 对齐一样，可以将它和最后一对窗口字符 $t_{i+m-1}t_{i+m}$ 对齐。在先前的例子中，每个窗口检验两个字符，平均平移 $4^2=16$ 个位置；通过考虑 3 个字符，每个窗口会检验 3 个字符，平均平移 $4^3=64$ 个位置。

通常，我们能够预处理 P，使得能够用窗口最后的 q 个字符来平移。每个窗口至少需要访问 q 个字符，然后平均平移 σ^q 个位置。q 的最佳值是什么？从先前的数字来看，似乎 q 越大越方便。然而，我们不可能平移多于 m 位，所以 $\sigma^q \leqslant m$ 是一个天然的限制。如果我们设置 $q=\log_\sigma m$，而且我们足够幸运，在读了 q 个字符后忽略了整个窗口，那么平均的搜索时间就是 $O(n\log_\sigma(m)/m)$。

实际上，这个平均的复杂度是最优的，我们选择的 q 也是接近正确的。可以分析说明，通过选择 $q=2\log_\sigma m$，平均的搜索时间可以达到最优的 $O(n\log_\sigma(m)/m)$。

agrep 软件工具使用了这个技术。选择一个散列函数，把 q-gram（长度为 q 的字符串）映射到一个整数范围。然后将从 P 的每个 q-gram 到 P 的末尾的距离记录在一个散列表中（冲突的解决方案是让更小的距离覆盖较大的）。对于不在 P 中的 q-gram，使用距离 $m-q-1$。现在 Horspool 算法几乎是逐字地运行的。

一个小的插曲是，因为最后一个 q-gram 很有可能与模式不匹配，所以我们尝试在验证窗口前就平移。为此，我们将 P 的最后一个 q-gram 加入到 d 的定义中，这样当窗口的最后一个 q-gram 与模式匹配时，就表示不需要平移。图 9-19 展示了它的伪代码。

```
Agrep (T = t1t2...tn, P = p1p2...pm, q, h( ), N)

for i ∈ [1,N] do d[i] ← m − q + 1
for j ← 0...m − q do d[h(p_{j+1}p_{j+2}...p_{j+q})] ← m − q − j
i ← 0
while i ≤ n − m do {
  s ← d[h(t_{i+m−q+1}t_{i+m−q+2}...t_{i+m})]
  if s > 0 then i ← i + s
  else {
    j ← 1
    while j ≤ m ∧ t_{i+j} = p_j do j ← j + 1
    if j > m then report an occurrence at text position i + 1
    i ← i + 1
  } }
```

图 9-19 在短字母表上匹配长模式的 agrep 算法的伪代码（简化版）。函数 $h: \sum^q \to [1,N]$ 散列 q-gram，它假设 $m \geqslant q$

9.5.2 复杂模式：自动机和位并行

Horspool 算法对于搜索简单模式来说工作得相当好。然而，在某些情况，模式会更复杂，如字符类、可选字符、通配符等。Horspool 算法和大多数传统的算法都无法很好地应用到复杂模式中。在本节，我们将介绍自动机（automata）和位并行（bit-parallelism）两种算法如何能够以一种简单、优雅和高效的方式处理许多这样的复杂模式。

1. 自动机

图 9-20 的顶部显示了一个搜索模式 $P=$ “**abracadabra**” 的非确定有限自动机（Non deterministic Finite Automation，NFA）。NFA 是一个有向图，其中，结点叫做状态（state），箭头叫做转移（transition），并用单个字符（$s \xrightarrow{c} s'$）或空字符串（$s \xrightarrow{\varepsilon} s'$，叫做 ε 转
375 移）标记。有时为了简便，我们将从状态 s 到状态 s' 的多个箭头折叠成多标签的单个箭头。

在图中无任何状态指向它的一个状态叫做开始状态，图中用双重圆圈标记的一个或多个状态叫做终结状态。NFA 根据如下的过程接受或者拒绝字符串，其中给定字符串作为输入：

1）NFA 启动时，只有开始状态是活跃的。

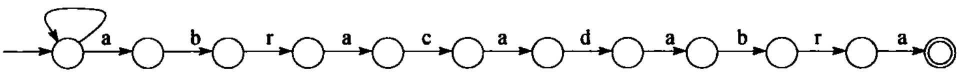
2）如果一个状态 s 是活跃的，并且有一个转移 $s\xrightarrow{\epsilon}s'$，那么状态 s' 也立刻被激活。更多的状态可以通过 ε 转移从 s' 激活，直到没有更多的状态被激活。

3）如果输入字符串耗尽了，那么这个过程就停止了。这时，当且仅当某个终结状态被激活，就称该 NFA 接受了这个字符串。

4）读取下一个输入字符 c。

5）新的活跃状态是这样一些 s'：当前存在活跃状态 s，而且在 NFA 中存在一个转移 $s\xrightarrow{c}s'$。

6）返回第 2 步。



B[a] = 0	0	0	1	0	1	0	1	0	0	1
B[b] = 1	1	0	0	0	0	0	0	1	0	0
B[r] = 1	0	1	0	0	0	0	0	0	1	0
B[c] = 1	0	0	0	1	0	0	0	0	0	0
B[d] = 1	0	0	0	0	0	1	0	0	0	0
B[*] = 1	0	0	0	0	0	0	0	0	0	0

图 9-20 搜索“**abracadabra**”的非确定性自动机和相应的表 B。开始的自环匹配任何字符。表中的每列对应于自动机的一条边

字符串被 NFA 接受的一个非操作性描述是：如果从开始状态到终结状态之间存在一条路径，并且连接所有转移上的标签字符串后就是 S，那么就说字符串 S 被一个 NFA 接受。

可以看出，在图 9-20 中的 NFA 接受任何以 P= “**abracadabra**” 结束的字符串。开始状态总是活跃的，因为那个自环可以接受任何字符。如果我们输入文本 T 给这个 NFA，那么它将会在 P 在 T 中的每个出现记录之后到达终结状态。注意，多个状态可以同时是活跃的。例如，在读取“abra”后，NFA 状态 0、1 和 4 都是活跃的。 376

一个有 m 个状态的 NFA，对于每个输入字符需要 $O(m)$ 时间运行，那么，在 T 中找到 P 的出现记录需要 $O(mn)$ 的时间。或者，它可以转换成一个确定有限自动机（Deterministic Finite Automation，DFA），它的转移用不同的单个字符来标记。DFA 每次总是最多只有一个状态是活跃的。把 NFA 转换成 DFA 有标准技术，能在 $O(n)$ 时间内高效地完成复杂模式的匹配（因为只需要追踪单个活跃的状态）。然而，对于很多复杂模式来说，这个构建过程虽然很强大，但不一定是必要的。在本节中，我们将说明，如何通过使用位并行技术高效地运行多个 NFA。至于更复杂的模式，采用 DFA 则是必要的，我们将在 9.5.4 节进行讨论。

2. 位并行和 Shift-And 算法

位并行利用机器字（如今，w=32 位、64 位或 128 位）内位操作固有的并行性。通过适当地使用这个性质，算法所执行的操作个数可以最多减少 w 倍，在实际中这是十分显著的。随后，我们将谈到位掩码（bit mask），它是保存在计算机寄存器中的位序列。位掩码从右向左读取，所以 $b_m\cdots b_1$ 的第一位是 b_1。位掩码的操作符包括：“|”表示按位或，“&”

表示按位与，“^”表示按位异或。一元操作符“~”对所有位取反。此外，“$mask \ll i$”表示将掩码中所有的位左移 i 位，并在右边补 0；“$mask \gg i$”是类似的。最后，可以像数字一样操作位掩码，如对它们进行加、减操作。

最简单的位并行算法叫做 Shift-And 算法，它能够匹配单个字符串。这个算法建立一个表 B，其中对每个字符保存一个位掩码 $b_m \cdots b_1$。当且仅当 $p_i = c$ 时，对 $B[c]$ 第 i 位的值置为 1（见图 9-20）。搜索的状态保存在一个机器字 $D = d_m \cdots d_1$ 中，其中当图 9-20 中的状态 i 活跃时，d_i 设置为 1。因此，当 $d_m = 1$ 时，表示找到了一个匹配。注意，状态 0 并未在 D 中，
377 因为它总是活跃的。

D 一开始设置为全 0，对于每个新的文本字符 t_i，算法在读取 t_i 后更新 D 来反映最新的活跃状态集合。图 9-21 给了这个算法的伪代码。比较有趣的是行（5），这行模拟了 NFA 的过程。不难把这个式子和 NFA 中的激活过程联系起来。首先，每个活跃状态尝试激活下一个状态（“$D \ll 1$”），而未显示的初始状态尝试激活状态 1（“|1”）。然而，只有由字符 t_i 标记的转移才能真正地激活目标状态，因此先前的操作的结果就是和 $B[t_i]$ 求“与”。

```
Shift-And (T = t1t2...tn, P = p1p2...pm)

(1)  for c ∈ Σ do B[c] ← 0
(2)  for j ← 1...m do B[pj] ← B[pj] | (1 << (j − 1))
(3)  D ← 0
(4)  for i ← 1...n do {
(5)     D ← ((D << 1) | 1) & B[ti]
(6)     if D & (1 << (m − 1)) ≠ 0
(7)     then report an occurrence at text position i − m + 1
     }
```

图 9-21 Shift-And 算法的伪代码

很容易看出，Shift-And 算法花费 $O(n)$ 时间，在实际应用中，考虑到要检验所有的文本字符，这个算法还是十分快的。然而，这个伪代码要求 $m \leqslant w$，也就是说，机器字必须有足够的位数来保证每个模式位置保存一位。对于长模式，最实用的方法就是搜索 $p_1 p_2 \cdots p_w$，然后直接对以此为前缀的记录验证整个 P。平均来说，这仍然是 $O(n)$ 时间。通过使用多个机器字来模拟 NFA，在最坏的情况下也能保证 $O(mn/w)$ 的时间复杂度。

3. 扩展 Shift-And 算法

对于简单字符串，Shift-And 算法无法与 Horspool 算法匹敌（除了在字符串十分短的情况下，Horspool 算法无法得到长的平移）。Shift-And 算法的优势在于，它能够（或者在扩展后）处理复杂得多的模式。最简单的例子是字符类（class of character）：P 中的一些位置表示某个字符集合，而不是单个字符。例如，当希望以大小写不敏感的方式搜索，或者希望查找整个单词（例如，当查找单词“**oil**”时避免找到“**spoil**”）。在后一情况下，我们可以用两个不包含字母的字符类来包围需要查询的词。当在最后位置附近搜索某个包含很大类的模式时，Horspool 算法的性能会极大地降低。相反，Shift-And 算法对这些困难完全不敏感。它只需要对每个属于 p_i 类的字符 c，将 $B[c]$ 的第 i 位设置为 1，而它的搜索算法和性能并未改变。

现在让我们考虑一个更加复杂的模式。假如我们搜索“**neighbour**”，但是希望“**u**”是
378 可选的，这样就可以同时找到这个词的英式写法和美式写法。图 9-22 展示了一个用 ε 转移实现这个功能的 NFA：一旦状态 7 被激活，不管是否读取“**u**”，状态 8 都能到达。

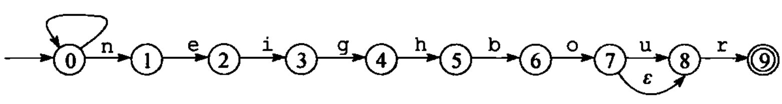


图 9-22 搜索“**neighbour**”，且“**u**”可选的非确定自动机

在复杂的模式中，自然语言搜索常常有用的另一个功能是通配符（wild card），或者更通用的可重复字符（或类）。这些模式位置可以在文本中连续出现一次或者多次。例如，我们可能希望通过查找“**transferred$ … to account #123456**”来得到银行中所有的转账记录，其中的“…”表示任何数字序列。又如另一个例子，我们查找“**well known**”，但是这两个词之间可能有连字符，一个或多个空格，或者是换行，如“**well known**”、“**well known**”、“**well-known**”、“**well--known**”、“**well - known**”和“**well \n known**”等。图 9-23 展示了解决最后这个例子的 NFA。

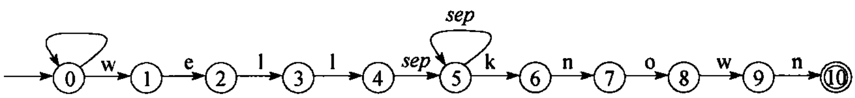


图 9-23　搜索“**well known**”，并且分割符可以是连字符、空格或者换行的任何序列的非确定自动机。符号“**sep**”代表集合 {—，‘ ’，\n}

字符的可选性、可重复性和类别可以结合起来，这样就有一个可选且可重复的类。图 9-24展示了一个处理所有这些情况的扩展 Shift-And 算法的伪代码。解释代码所有的细节并不在本章的范围之内，因为这只是所有可以处理的模式类型的一个例子。然而，弄清这段伪代码如何工作，并以此推广到其他情况，对读者来说是一个很有趣的练习。9.8 节引用了一些更深入的研究。

```
Shift-And-Extended (T = t1t2...tn, m, B[ ], A, S)

I ← (A >> 1) & (A ^ (A >> 1))
F ← A & (A ^ (A >> 1))
D ← 0
for i ← 1...n do {
   D ← (((D << 1) | 1) | (D & S)) & B[ti]
   Df ← D | F
   D ← D | (A & ((~ (Df − I)) ^ Df))
   if D & (1 << (m − 1)) ≠ 0
   then report an occurrence at text position i − m + 1
}
```

图 9-24　一个处理字符类、可选、可重复字符的扩展 Shift-And 算法的伪代码。出于简化目的，已经预先对字符和字符类计算好了表 B（忽略可选或可重复的）。位掩码 A 和 S 分别表示可选和可重复的位置。同样出于简化目的，我们假设模式中的第一个和最后一个位置并不是可选的（否则可以简单地移除它们）

9.5.3　更快的位并行算法

在 Horspool 等用于简单字符串匹配的快速算法和 Shift-And 等用于复杂模式匹配的较慢的算法之间，我们似乎必须做出选择。然而，有一些位并行算法，既可以处理复杂模式，又能够跳过文本字符。然而，随着模式变得越来越复杂，那些算法会渐渐变慢。 379

1. 后缀自动机

模式 P 的*后缀自动机*（suffix automaton）是一个识别 P 所有后缀的自动机。这个自动机的非确定版本有一个十分规则的结构，如图 9-25 所示。

BDM 算法基于后缀自动机实现。我们将展示一个简化的变种（它在自然语言文本上更快）。为了搜索模式 P，需要建立 P^{rev}（反向模式）的后缀自动机。这个算法反向（从右往左）扫描文本窗口，并把字符提供给 P^{rev} 的后缀自动机。如果在扫描了 $t_{i+m}t_{i+m-1}\cdots t_{i+j}$ 后，自动机耗尽了活跃状态，那么表示 $t_{i+j}t_{i+j+1}\cdots t_{i+m}$ 并不是 P 的一个子串。因此，没有任何 P 的实例能够包含这个文本子串，这个窗口可以很安全地移过 t_{i+j}。相反，如果到达了窗口的开始，自动机仍然有活跃的状态，那么表示这个窗口与模式匹配（见图 9-25），所以我们输出这个记录，并移动这个窗口一个位置。

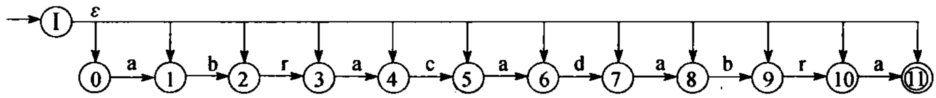


图 9-25 关于 $P=$ “**abracadabra**”的非确定后缀自动机

这个算法在最坏情况下的时间复杂度是 $O(mn)$，但平均是 $O(n\log_\sigma(m)/m)$，也是最优的。因为需要实现确定性的后缀自动机，所以这使得 BDM 比我们先前看过的一些算法要更
380 复杂。一个比较吸引人的变种叫做 BNDM，它采用位并行实现了后缀自动机，并且当模式并不十分长时（最多是机器字长的两倍）能取得更好的性能。

BNDM 的伪代码如图 9-26 所示。行（8）将所有的 ε 转移从状态 I 传播到每一个其他状态，并将窗口的最后一个字符输入 NFA。然后，扫描窗口，直到完全遍历它，或者自动机耗尽了所有的活跃状态。对于第一种情况，在行（10）输出该记录；对于这两种情况，窗口都会平移过扫描结束的位置，如行（11）所示。

```
BNDM (T = t1t2...tn, P = p1p2...pm)

(1)  for c ∈ Σ do B[c] ← 0
(2)  for j ← 1...m do B[pj] ← B[pj] | (1 << (m − j))
(3)  i ← 0
(4)  while i ≤ n − m do {
(5)     j ← m − 1
(6)     D ← B[t_{i+m}]
(7)     while j > 0 ∧ D ≠ 0 do
(8)        D ← (D << 1) & B[t_{i+j}]
(9)        j ← j − 1
        }
(10)    if D ≠ 0 then report an occurrence at text position i + 1
(11)    i ← i + j + 1
     }
```

图 9-26 BNDM 算法的伪代码

BNDM 算法可以很容易地应用于复杂模式。给定任何如图 9-22 或图 9-23所示的 NFA，我们几乎可以自动地获得相应的类 BNDM 搜索算法。9.8 节给出了一些包含更多细节的文献。

2. 交错 Shift-And 算法

另一个获得最优平均搜索时间的想法是从 q 个文本字符中读取一个。详细来说，假设 $P=$ “**neighborhood**”和 $q=3$，如果我们从 3 个文本位置中读取一个，而且 P 出现在某个文本窗口 $t_{i+1}t_{i+2}\cdots t_{i+m}$，那么，根据 i 模 3 的值，我们将在窗口中读到“**ngoo**”、“**ehro**”和“**ibhd**”。因此，可以并发搜索 P 的这三个子序列，这样保证 P 的每个记录都能检测出。

这个并发搜索可以通过运行$\lfloor m/q \rfloor$个交错（interlaced）搜索来实现，这些交错搜索使用与 Shift-And 算法一样的位并行方法。现在，开始状态可以激活 P 的前 q 个位置，并且位并行的移位是每次 q 个位置。图 9-27 显示了这个自动机，图 9-28 是它的伪代码。注意，找到这三个子序列中的一个并不能保证我们已经找到了 P，所以需要在行（8）进行一次 Shift-
381 And 验证。

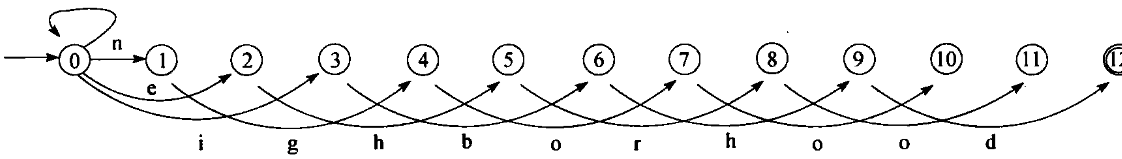


图9-27 用于 $P=$ “**neighborhood**”、$q=3$ 的交错搜索的非确定后缀自动机

不难看出，如果令 $q=m/(2\log_\sigma m)$，那么这个算法的平均搜索时间是最优的，为 $O(n\log_\sigma(m)/m)$（假设 $m\leqslant w$）。然而，并不清楚图 9-22 和图 9-23 中更复杂的模式是否能用这个技术处理。

9.5.4 正则表达式

上面讨论的更复杂的模式是正则表达式的特例。一般的正则表达式（见 7.1.2 节）不能如图 9-22 和图 9-23 中那样方便地顺序排列，而是需要一个更复杂的应对方法。

```
Interlaced-Shift-And (T = t_1 t_2 ... t_n, P = p_1 p_2 ... p_m, q)

for c ∈ Σ do B[c] ← 0
for j ← 1...m do B[p_j] ← B[p_j] | (1 << (j − 1))
S ← (1 << q) − 1
D ← 0
for i ← 1...⌊n/q⌋ do {
   D ← ((D << q) | S) & B[t_{q·i}]
   if D & (S << (⌊m/q⌋ · q − q)) ≠ 0
   then run Shift-And over t_{q·i−m+1} ... t_{q·i+q−1}
}
```

图 9-28 采样间距为 q 的交错 Shift-And 算法的伪代码（简化版）。假设 $m \geqslant q$

处理正则表达式的第一部分就是建立它的 NFA。有不同的 NFA 构建方法，我们选择更为传统的 Thompson 技术加以说明，因为它比较容易解释。图 9-29 描述了这个递归的构建方法。

一旦 NFA 建立完毕，我们在开始状态加一个自环（接受任何字符），然后这个自动机就可用于搜索了。如果正则表达式有 m 个符号，那么 Thompson 构建方法就会生成一个大小为 $O(m)$ 的自动机。搜索可以在 $O(mn)$ 时间内完成。

可以将 NFA 确定化，转换成 DFA。基本方法就是把 NFA 状态的每个子集当做一个 DFA 状态，并且满足下面的性质：在接受一个字符串后，DFA 所在的状态代表 NFA 的状态集合 $\{s_1, \cdots, s_r\}$ 当且仅当 NFA 转移到 $\{s_1, \cdots, s_r\}$ 中的一个状态。

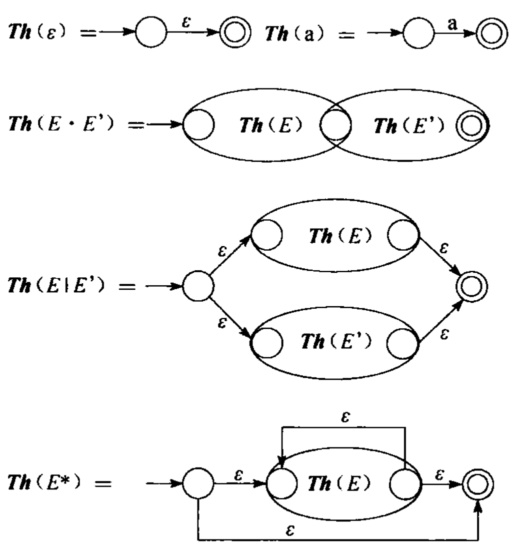


图 9-29 从正则表达式到 NFA 的递归 Thompson 构建方法。每个 NFA 只有一个终结状态 382

定义 ε-closure（s）是从 s 开始，通过 0 个或多个 ε 转移可达的 NFA 状态集合。DFA 开始的状态是 ε-closure（I），其中 I 是 NFA 的开始状态。然后，令 S 是一个已经生成的 DFA 状态。从 S 通过字符 c 可达的状态 S' 是：

$$S' = \bigcup_{s \in S, s \xrightarrow{c} s'} \varepsilon\text{-closure}(s') \qquad (9\text{-}1)$$

我们先产生 I，然后必须生成所有从 I 开始、通过每个可能的字符 c 可达的状态。每一个新的状态都必须轮流处理，以此来生成从它们开始、通过任一字符可达的状态。这个过程一直持续到所有新生成的状态都处理完毕，然后 DFA 就构建完毕。DFA 的终结状态是那些包含某些 NFA 终结状态的集合。

在最坏的情况下，DFA 可能包含多达 $2^{O(m)}$ 个状态，但是，在许多实际的情况中，它要小得多。有些系统化的步骤来最小化 DFA 的状态数目，而不影响它接受的字符串集合。

一旦得到了这个 DFA，那么对于原来的正则表达式，可以在 $O(n)$ 时间内扫描整个文本：我们从 DFA 的开始状态开始，依次读入每个文本字符，移到新的状态。每当到达了终结状态，就输出这个记录。每个转移可以在常数时间内完成，只要预先计算一个状态转移表，它对于每个 DFA 状态和字符，给出新的 DFA 状态。 383

图 9-30 展示了将 Thompson 构建方法应用到正则表达式时生成的 NFA，以及将上述过程应用到 NFA 所生成的 DFA 和最小化的 DFA。

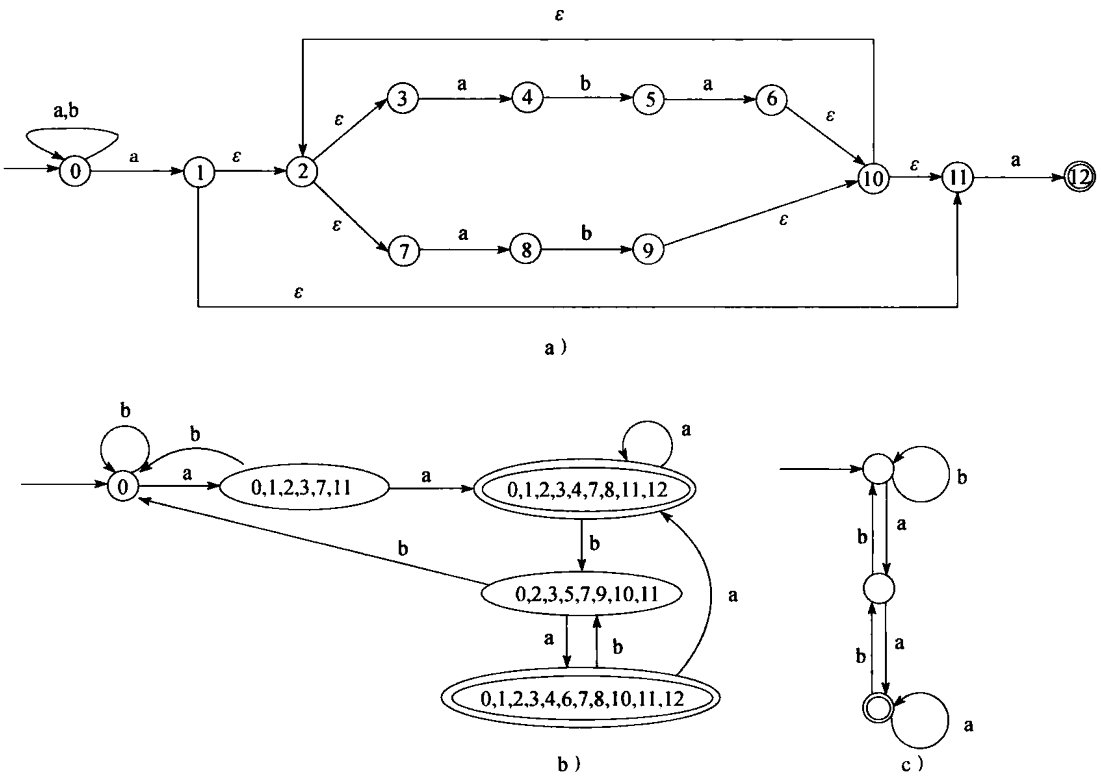


图 9-30 a）Thompson 方法的 NFA；b）对应的 DFA；c）最小化 DFA，其中的正则表达式为“**a**（**aba**|**ab**）$*$**a**”

最近，研究人员提出了使用位并行方法来避免构建确定自动机。与经典方法相比较的结果各有优劣，每一个都有比另一个好的情况。我们上面已经展示的经典方法对出现在信息检索应用中的正则表达式更加有效，而对于出现在计算生物学场景中的更大的正则表达式，可能使用位并行方法会更好。

9.5.5 多重模式

单个字符串匹配算法（参见 9.5.1 节）可以扩展成处理多字符串$\mathcal{P}=\{P_1, P_2, \cdots, P_r\}$。例如，当只搜索少量模式时，我们可以扩展 Horspool 算法，使得 $d[c]$ 表示单个模式 P_i 的 $d_i[c]$ 值中的最小值，如图 9-17 所示，用聚合的 d 函数进行搜索。为了计算每个 d_i，我们必须以$\mathcal{P}$中最短模式的长度 m 截断 P_i，并且必须改变图 9-18 的行（6），我们不再比较
384 t_{i+j}和 p_j，而是将后续的符号 t_{i+j}加入到在$\mathcal{P}$中所有模式上建立的 trie 树中（这个 trie 树中的模式并未被截断，因此验证过程可能会读取当前窗口之外的文本符号）。

当查找更多的模式时，上面的方案就必须扩展成根据它最后的 q-gram 平移窗口。这恰恰就是 agrep 工具对于多重模式匹配所采用的方案，而且它在实际中性能非常好。q 的值必须是 $2\log_\sigma m+\log_\sigma r$，这样保证平均及最优搜索时间是 $O(n\log_\sigma(mr)/m)$。其他性能较好的变种是对 BDM 的一些扩展。

当模式的数量相对于它们的长度十分大时，没有任何字符跳过算法表现得比较好，所以

保证每个文本字符只扫描一次的方案是更可取的。位并行算法（见 9.5.2 节）在这个情况中也没有用。相反，我们可以建立能识别$\mathcal{P}$：$P_1 | P_2 | \cdots | P_r$ 的正则表达式，然后运用 9.5.4 节描述的 DFA 构建算法。生成的 DFA 的大小与所有模式的长度之和成比例，搜索时间是 $O(n)$。这个方案与经典的用于多字符串匹配的 Aho-Corasick 算法十分类似。

注意，这个技术也能扩展成搜索多重复杂模式，然而这次生成的 DFA 的大小可能与模式的长度成指数关系。

9.5.6　近似搜索

近似字符串匹配用一个阈值参数 k 扩展简单字符串匹配，不只是精确的 P 的记录需要输出，而且也要输出那些最多有 k 个“错误”的 P 的记录。对于“错误”，有不同的定义。一个十分流行的定义对应于所谓的 Levenshtein 距离或编辑距离：“错误”是指删除、插入或替换一个字符（见 6.5.3 节）。这个模型足够简单，可以执行快速搜索，并且对大多数信息检索场景是有用的。我们将要说明有些算法可以通过很小的改变就应用于其他错误模型，包括交换相邻的字符或者给不同的替换赋予不同的权重。

该问题可以一般化为近似模式匹配：近似地搜索一个表示某个字符串集合$\mathcal{P}$的复杂模式或正则表达式，输出任何能够在最多 k 个错误的情况下转换成$\mathcal{P}$的一个字符串的文本子串。

对这个问题有许多解决方案，大多数只解决近似字符串匹配。我们只描述最实用的一些方法。

1. 动态规划

近似字符串匹配的一个经典的解决方案是基于动态规划。矩阵 $C[0..m, 0..n]$ 一列接着一列填充，其中 $C[i, j]$ 表示将 $p_1p_2\cdots p_i$ 与 $t_1t_2\cdots t_j$ 的某个后缀相匹配所需要的最小错误个数。其计算方式如下： 385

$$
\begin{aligned}
&C[0,j]=0,\\
&C[i,0]=i,\\
&C[i,j]=\text{if}(p_i=t_j)\ \text{then}\ C[i-1,j-1]\\
&\qquad\qquad \text{else}\ 1+\min(C[i-1,j],C[i,j-1],C[i-1,j-1])
\end{aligned}
\tag{9-2}
$$

在那些 $C[m, j] \leqslant k$ 的文本位置 j，输出匹配。这对应于近似记录的末尾位置。

图 9-31 展示了这个算法。它需要 $O(mn)$ 的时间。因为当计算新的一列时，只有矩阵的前一列是需要的，所以这个算法可以只占用 $O(m)$ 空间。

		k	o	l	o	r	a	m	a
	0	0	0	0	0	0	0	0	0
c	1	1	1	1	1	1	1	1	1
o	2	2	1	2	1	2	2	2	2
l	3	3	2	1	2	2	3	3	3
o	4	4	3	2	1	2	3	4	4
u	5	5	4	3	2	2	3	4	5
r	6	6	5	4	3	**2**	3	4	5

图 9-31　在文本“**kolorama**”中允许带有两个错误地搜索“**colour**”的动态规划算法。加粗的条目表示一个近似记录结束的位置

这个算法尽管并不十分快，但是可以很灵活地适应其他代价函数。通过利用动态规划矩阵的某些性质，它的几个扩展算法在最坏的情况下取得了 $O(kn)$ 的时间复杂度，平均情况下甚至更少。还能得到一个简单的 $O(kn)$ 平均时间复杂度的算法，只要在计算每列的值时，如果知道所有接下来的值肯定会超过 k，那么就不计算下去了。图 9-32给出了这个变种的优化伪代码。

2. 自动机和位并行

近似字符串匹配也可以表达为一个 NFA 搜索。考虑图 9-33 所示的允许 2 个错误的 NFA。每行表示已经看到的错误个数，第一行是 0，第二行是 1……每一列表示匹配给定位置的模式。在每次迭代中，先读取一个新的文本字符，然后自动机改变其活跃状态。水平箭头表示匹配一个字符，垂直箭头表示插入“错误”，实线对角箭头表示替代“错误”，虚线对

角箭头表示删除“错误”(它们是 ε 转移)。当最右下角的状态被激活时,就表示自动机接受了某个文本位置作为允许 k 个错误的匹配的末尾。

```
Approximate-DP (T = t1t2...tn, P = p1p2...pm, k)

 (1)  for i ← 0...m do C[i] ← i
 (2)  last ← k+1
 (3)  for j ← 1...n do {
 (4)    pC, nC ← 0
 (5)    for i ← 1...last do {
 (6)      if p_i = t_j then nC ← pC
 (7)      else {
 (8)        if pC < nC then nC ← pC
 (9)        if C[i] < nC then nC ← C[i]
(10)        nC ← nC + 1
          }
(11)      pC ← C[i]
(12)      C[i] ← nC
        }
(13)    if nC ≤ k
(14)    then if last = m then report an occurrence ending at position i
(15)         else last ← last + 1
(16)    else while C[last − 1] > k do last ← last − 1
      }
```

图 9-32 基于近似字符串匹配的动态规划伪代码,做了一些优化,平均时间复杂度是 $O(kn)$。它假设 $k<m$

不难看出,每次自动机中的一个状态活跃时,所有同列且行数更大的状态也是活跃的。
386 而且,在某个给定的文本字符,如果我们在每一列中收集最小的活跃状态行号,那么正好得到动态规划算法中当前列的值(不区分大于 k 的值)。比较图 9-33 和图 9-31。

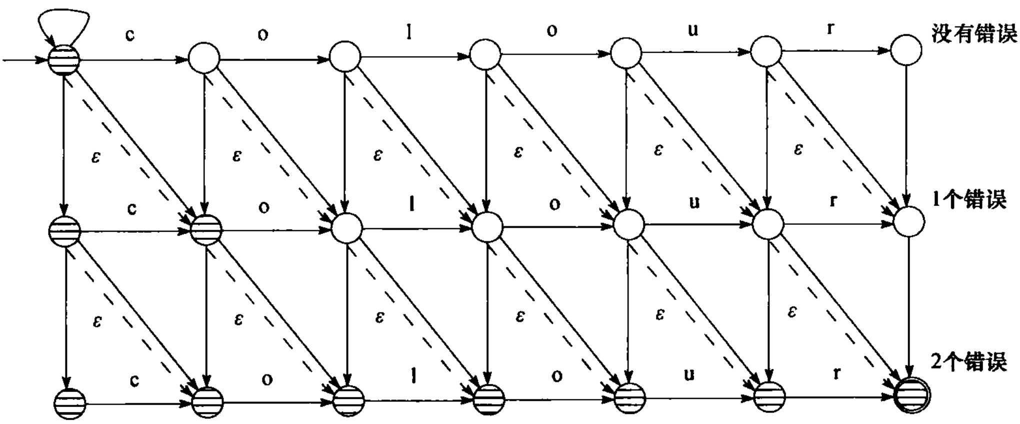


图 9-33 对允许带有两个错误的模式“**colour**”进行近似字符串匹配的 NFA。有阴影的状态
387 是那些在读取文本“**kolor**”后的活跃状态。未标记的转移匹配任何字符,虚线的箭头是 ε 转移

如 9.5.4 节所描述的那样,也可以使自动机确定化(DFA)。尽管搜索阶段是 $O(n)$ 复杂度,但 DFA 在这种情况下可能会变得十分大。一个更好的方案是基于位并行的扩展 Shift-And 算法。需要注意的是,该 NFA 的每一行几乎都是图 9-20 的 NFA 的完全复制。因此,当使用垂直和对角箭头时,可以使用 $k+1$ 个 Shift-And 过程来实现。

图 9-34 给出了这个伪代码。我们感兴趣的是行 (8)。除了正常的 Shift-And 操作(除了 NFA 的第一行之外,都不包含自环)外,我们还模拟了垂直的箭头,即与上一轮迭代中的

D_{i-1}的所有值为“1”的位进行“或”操作（上一轮的 D_{i-1}值保存在 pD 中）。类似地，对角的箭头用“(pD<<1) |1”来模拟，ε 转移（同样是对角的）对应于“(nD<<1) |1”。在这种情况下，我们使用新的 D_{i-1}值，因为 ε 转移是即刻传递的。

如果 $m \leqslant w$，那么位并行算法需要 $O(kn)$ 的时间复杂度。通过模拟更长的机器字，更长的模式可以在 $O(kmn/w)$ 时间内处理。位并行同样可以用于并行计算动态规划矩阵，平均的时间复杂度是$O(kn/w)$，最坏的时间复杂度是$O(mn/w)$。对于短模式，这个算法最坏的时间复杂度是 $O(n)$，在实际应用中是十分快的。不过，我们已经展示了更加灵活的算法：为了搜索任何允许错误的复杂模式，如果我们能够用位并行模拟精确搜索，那么我们能够很容易地安排$k+1$个这样的模拟，以获得对应近似模式匹配问题的位并行算法。关于算法细节，将在 9.8 节中给出一些条目。

```
Approximate-Shift-And (T = t1t2...tn, P = p1p2...pm, k)

for c ∈ Σ do B[c] ← 0
for j ← 1...m do B[pj] ← B[pj] | (1 << (j − 1))
for i ← 0...k do Di ← (1 << i) − 1
for j ← 1...n do {
  pD ← D0
  nD, D0 ← ((D0 << 1) | 1) & B[ti]
  for i ← 1...k do
    nD ← ((Di << 1) & B[ti]) | pD | ((pD | nD) << 1) | 1
    pD ← Di, Di ← nD
  if nD & (1 << (m − 1)) ≠ 0
  then report an occurrence ending at position i
}
```

图 9-34　使用 Shift-And 算法的近似字符串匹配算法的伪代码

3. 筛选

通常，确定一个文本位置无法匹配比确定它能在 k 个错误内匹配要简单多了。筛选（filtration）是基于在文本上进行快速的过滤，以期望舍弃大部分文本位置，然后在那些无法舍弃的区域上应用另一个近似搜索算法。筛选的效果在特定的 k/m 限制范围内都很好。 388

一个简单快速的过滤过程是：将模式切割成 $k+1$ 个长度相同的片段。事实上，P 的任何近似记录都必须完全相同地包含这些片段中的至少一个（因为每个错误至多只能改变一个这样的片段）。然后，我们可以对这些片段运行一个多模式的搜索算法（见 9.5.5 节）。如果片段 $p_j \cdots p_{j'}$ 出现在 $t_i \cdots t_{i'}$，那么我们在 $t_{i-j+1-k} \cdots j_{i-j+m+k}$上运行近似字符串匹配算法。对于 $k<1/\log_\sigma m$，算法平均花费 $O(nk\log_\sigma(m)/m)$ 的时间。

另一个更加强大的算法继承于 BDM（见 9.5.3 节），它在实际应用中也表现得十分好。选择一个小的值 q，对所有 σ^q 个可能的 q-gram 中的任意一个，在 P 中查找 q-gram S，然后在表 $D[S]$ 中记录将 S 匹配于 P 中所需的最小的错误数。现在，在 T 上滑动窗口 $t_{i+1} \cdots t_{i+m-k}$。反向读取窗口的 q-gram：$S_1 = t_{i+m-k-q+1} \cdots t_{i+m-k}$，$S_2 = t_{i+m-k-2q+1} \cdots t_{i+m-k-q}$等。累加 $D[S_1]+D[S_2]+\cdots+D[S_r]$，直到总值超过 k。此时，我们知道 $S_r \cdots S_2 S_1$ 不可能包含在 P 的任何记录中，因此我们可以安全地移动窗口，从 S_r 的第二个字符开始，即 $i \leftarrow i+m-k-rq+1$。如果我们读取窗口的所有整个 q-gram，并且错误总值仍然不超过 k，那么我们必须显式地检查从窗口位置开始的记录，然后移动窗口一个位置。

9.5.7　搜索压缩文本

在 6.8 节，我们已经描述了适合文本数据库的压缩机制，但仅关注减少空间。然而，我们已经注意到搜索压缩文本的可能性，甚至介绍了一些编码方法（如字节霍夫曼和密集编码），它们的主要优点是能够直接搜索压缩文本。其基本原理是在之前介绍过的字节霍夫曼编码或密集编码上搜索匹配单个词的模式。

我们首先重新考虑在压缩文本上进行序列搜索，展示信息检索领域所关注的更加复杂的

搜索如何在 6.8 节所提出的格式上进行。我们已经展示了在文本中搜索与单个词或者复杂模式相匹配的词串的过程，即标记匹配模式的词汇表条目，然后遍历压缩文本，识别原文中连续的符号，检查它们是否被标记过。对该简单方法进行扩展，将它与本章介绍的位并行技术结合起来，可以得到一个更强大的方法，能够匹配复杂得多的模式。

让我们从短语查询开始。6.8 节提到，搜索一个短语可以通过压缩每个词，并在压缩的文本中搜索目标符号连接成的字符串来实现。这是正确的，只要满足下面的条件：1）短语由简单的词组成，每个词都可以转换为一个码字；2）我们希望分隔符的形式与查询中的完全一样。也就是说，如果我们用这个技术查找 **"United States"**，并且对这些词的编码分别是
389 C_{United}和 C_{States}，那么对字符串 $C_{\text{United}}\ C_{\text{States}}$的搜索就不会找到用换行符等其他分隔符分隔的短语，因为这会被转换为码字 $C_{\text{United}}C_{\backslash n}C_{\text{States}}$。回想一下，在无空格单词模型中，并不对一个单空格的分隔符进行编码，然而却对任何其他的分隔符分配码字。

一个更加健壮的搜索方案是定义*词汇模式*（word pattern）。例如，我们可能希望搜索任何以大小写不敏感的形式，并且允许带有两个错误匹配 **"United"**，然后是一个分隔符，之后是任何以大小写不敏感的形式，并且允许带有两个错误匹配 **"States"**。为了找出词语被单个空格或者任何其他序列分隔的情况，我们必须在词之间允许任何分隔符号，而且这个符号在我们的搜索中必须是可选的。

这个搜索问题可以通过一种叫做*码字上的自动机*（automaton over codeword）的方法建模。令 C 是由压缩器创建的不同码字的集合。如果我们把 C 当做一个字母表，把压缩文本看做由 C 上的原子符号组成的序列，那么可以将搜索问题定义成找出某个复杂模式的记录，如 9.5.2 节所述。我们的模式有三个状态，每个状态表示一类字符：第一个是码字（C 的元素）的集合，这些码字对应于以大小写不敏感的形式，并且允许带有两个错误匹配 **"United"**；第二个是分隔符的码字的集合，它是一个可选的类；第三个与第一个类似，但对应于词 **"States"**，如图 9-35 所示。注意它与图 9-20 和图 9-22 的相似度。我们对词汇表中的每个词和分隔符建立一个位掩码，这些位掩码的数组是用于位并行搜索的 $B[]$ 表。我们考虑短

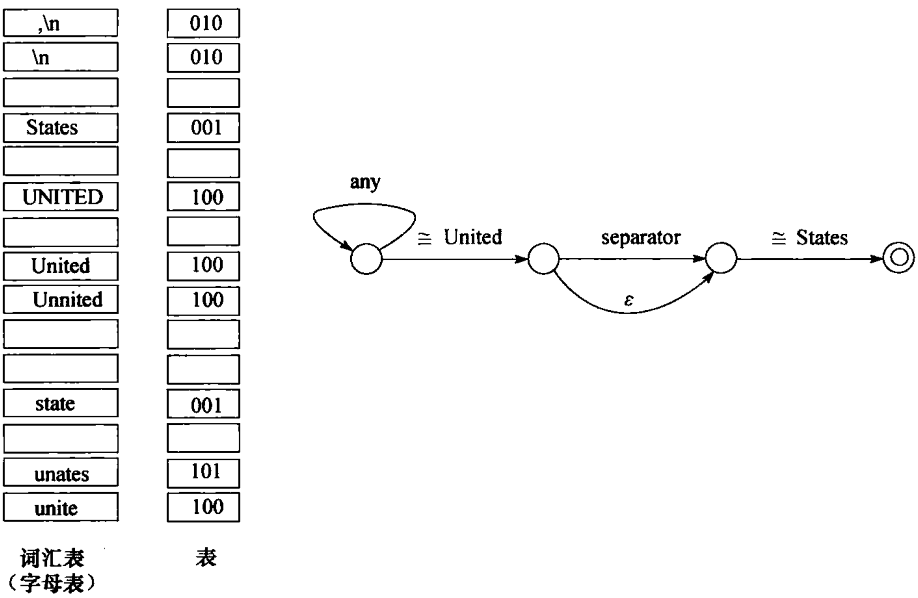


图 9-35 以大小写不敏感的形式搜索短语 **"United States"** 的通用方案，其中每个词允许有两个错误，词之间允许任何分隔符

语的元素，包括分隔符。对于第 i 个元素，我们顺序遍历词汇表，设置或清除每个位掩码的第 i 位。然后，我们建立一个 NFA，并用位并行来模拟它。在我们的例子中，之前已经解释过，第二个位置必须是可选的。这个搜索就等同于 **Shift-And-Extended** 过程（见图 9-24）的一个变种，其中的 B 表我们已经建立，同时参数 $A=010$，$S=000$。

这个搜索算法与图 9-24 中代码的不同之处在于：它并不是直接在字符上进行操作，相反，算法中 **for** 循环的每次迭代处理文本的一个完整的词或者分隔符。在执行行（5）前，我们必须得到下一个码字的 $B[]$ 的值（即当我们把压缩文本看做 C 上的一个序列的下一个符号）。为此，我们从压缩文本中读取尽可能多的字节，直到我们识别出下一个码字，而且根据码字我们可以得到对应的词汇表中的词或分隔符。从那个词或分隔符，我们得到它的 $B[]$ 的值，然后用行（5）进行处理。

这个过程可以用于搜索复杂得多的模式，包括一些在非压缩文本中也很难执行的搜索任务。例如，假设我们希望搜索 **"the number of elements successfully classified"**，或者其他类似的短语。许多其他的短语在某种程度上表示同样的意思，例如：

```
the number of elements classified with success
the elements successfully classified
the number of elements we successfully classified
the number of elements that were successfully classified
the number of elements correctly classified
the number of elements we could correctly classify
...
```
390

可以看出，尽管通过允许近似匹配单个词，可以匹配某些变体，但是我们需要一个更加强大的机制来得到那些语言学上的变化。如 9.5.6 节所说的，字符级别的近似字符串匹配可以从本地的错误中恢复，如那些因为输入错或拼错而产生的错误。为了从上面所说的语言学上的变体中恢复，我们必须采用词级别的近似字符串匹配。在这个模型中，我们搜索词的序列，并允许一定数目的丢失、新增或替换的词。例如，在允许三个词级别的错误时，可以将上面例子中所有的变体恢复出来（假设我们同时允许字符级别的错误把 **"classify"** 转换为 **"classified"**）。

通过扩展图 9-35 所说明的方案，可以很容易地得到词级别的近似字符串匹配。想象这次我们使用一个像图 9-33 一样的 NFA，其中每列对应于短语中的一个词。我们可以模拟 **Approximate-Shift-And**（见图 9-34）过程，其中，每个后续的 $B[]$ 值可以在像之前那样，通过解析一个码字后得到。

9.6　多维索引

在多媒体数据中，我们可以用多个数字特征代表每个对象。例如，从一张图我们可以抽取出颜色直方图、边缘位置等。在这种情况下进行搜索的一种方法是，把这些对象特征映射到多维空间中的点，然后使用多属性访问方法（也可以叫做*空间访问方法*（Spatial Access Method，SAM））来对它们进行聚类和搜索。另一个方法是为对象定义一个距离函数，然后使用基于距离的索引（也叫做*度量访问方法*（Metric Access Method，MAM））。 391

主要映射方法可以分为三类：1）R^* 树和 R 树家族的其余成员；2）线性四叉树；3）网格文件。

这些方法中的一些方法随着维度的增加产生指数爆炸现象，最终变成顺序扫描。线性四叉树的代价与查询区域的超曲面成比例 [545]；而超曲面随着维度呈指数级数增长。网格文件面临类似的问题，因为它们需要一个目录，这个目录也随着维度呈指数级数增

长。如果能保证R树结点的扇出均大于2的话，那么基于R树的方法对于高维数据似乎是最健壮的方法。下面，我们简单地介绍R树和它的变种，因为它是空间访问方法的一个典型形式。

R树通过最小边界矩形（Minimum Bounding Rectangle，MBR）来表示一个空间对象。数据矩形组合起来形成父结点，父结点又递归地组合起来，形成祖父结点，最终形成一棵树。一个父结点的MBR完全包含子结点的MBR；MBR之间允许重叠。树的结点对应于硬盘页。硬盘页，或者叫“硬盘块”，是在硬盘表面连续的字节，通常只需要一次硬盘访问就可以得到。在树中插入、分割和删除操作的目标是形成好的簇，即少量、紧密的父MBR。图9-36展示了数据矩形（黑色），并形成一个扇出为3的R树。图9-37展示了该R树的文件结构，其中结点对应于硬盘页。

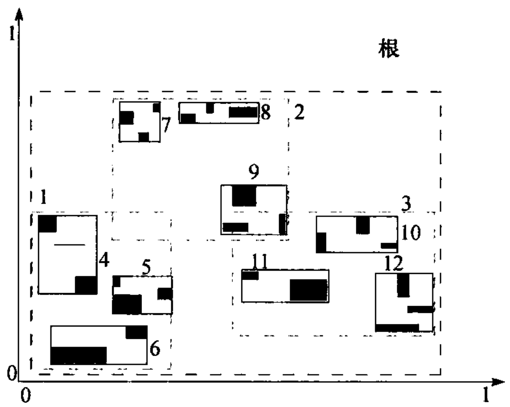


图9-36 扇出为3的R树的数据（黑色矩形）。实线、轻虚线、重虚线表示父亲、祖父和曾祖父结点（在这个例子中是根结点）

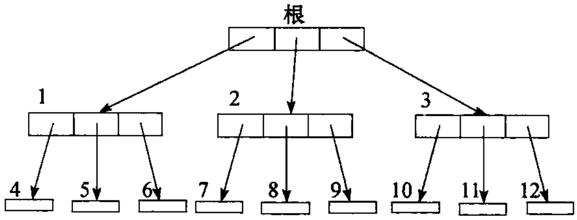


图9-37 图9-36中的R树（扇出为3）的文件结构

范围查询指定了某个兴趣区域，需要检索与之相交的所有数据区域。为了回答这个查询，首先找出满足条件的数据区域的超集：计算查询区域的MBR，然后递归地向下遍历R树，排除那些其MBR与查询MBR不相交的分支。因此，R树将使我们快速地找到那些其
392 MBR与查询MBR相交的数据区域。然后进一步检查得到的数据区域与查询区域之间的交集。

最初关于R树的论文激发了大量后续的工作，如在9.8节所描述的。在众多变种中，R^*树[165]似乎是性能最好的方法之一。R^*树在结构上与R树一样；主要区别是在分割算法上巧妙的提高，它基于一个称为*强制重新插入*（forced reinsert）概念。当一个结点溢出时，小心地选择它的一些子结点；把它们删除，然后重插入，通常会生成一个拥有更好结构的R树。然而，需要特别关注的是，任何空间访问方法都可以使用，如X树、SR树等。

当从对象中抽取数字特征并不简单，或者对象之间存在熟知的距离函数时，也可以使用度量访问方法。例如，编辑距离可以用来衡量字符串之间的相似度，余弦距离或相似性可以用来衡量文档间的相似度（见6.5.3节）。在度量空间中，一个舍弃对象的主要技术是三角不等式。我们在本章的最后给出了这种数据结构的参考文献。

9.7 趋势和研究问题

本章关注信息检索系统的实现，主要关注效率问题。我们介绍了索引搜索和序列搜索；在实际的实现中，很多情况下这两种搜索是结合的。我们考虑了不同的检索模型，如排序检

索、布尔检索、结构化检索、面向词语的全文搜索，以及面向任意文本子串的全文搜索。我们还介绍了压缩机制，以及它们与索引和序列搜索之间的关系。

在索引和搜索文本数据库领域，如今的主要趋势是：

- **大文档集**。可用的电子文档数据的数目增长很快。现在，通常需要处理上数吉字节(GB)的文本，并且必须每秒回答多个查询。这对信息检索系统提出了巨大的挑战。即使是十分简单的结构，如倒排索引，也需要复杂的实现以便在非常大的数据库上得到可观的性能。问题通常不是存储空间，而是二级存储设备的速度。处理器和相对较慢的外部设备的速度导致新的空间与时间的折中，而在几年前是不推荐这样做的。例如，为了避免访问外部存储，直接操作压缩形式的索引是比较值得的，即使这比不压缩的操作要更慢。同样，在能够取得好效果的复杂检索模型与高效地实现 393
它们之间会有冲突。
- **复杂搜索**。随着文本数据库的增长，并变得越来越混杂和易出错，需要有更强的查询工具来实现有效的检索。一方面，容错搜索或复杂模式匹配等一些特性可以提高召回率。另一方面，利用文本结构的查询、长的合取查询，或者短语查询，也可以提高准确率。由于这些原因，高效地支持一些不只是简单词语的短序列的查询就显得至关重要。
- **压缩索引**。因为中央处理器和外部设备的性能差别不断增大，以及这个领域最新的发展，文本检索和压缩不再认为是两个无关的事情。相对于不压缩的文本搜索，直接在压缩文本上的搜索，同时提供了更好的（有时甚至是好得多）时间性能和更少的空间开销。最近，有很多关于压缩和索引之间关系的研究，发现了一些重要的联系，由此可以产生一些很实用的自索引结构。这些索引比文本占用更少的空间，能够以索引的方式访问它，而且，它们还可以替代文本，因为它们可以重现任何文本子串。这方面的研究在未来的几年内有希望得到更进一步的结果。
- **优化处理**。新的计算机体系结构有多个处理器，它们可能是完全一样的，可以方便地实现并行化；或者是不一样的，如在高级的多核 CPU 中。如何充分利用可能是不同类型的多个处理器将变得越来越重要。一个特别的例子是，利用当前机器上的可用图形处理器 [500]。
- **在高维空间中搜索**。直到今天，尽管有很多的研究，但在高维空间中搜索对象时，可扩展性仍然是一个问题。这就是所谓的*维度灾难*（curse of dimensionality）。然而，通常对于上百万的对象，搜索时间是可接受的，但是对于 Web 上数十亿的对象却仍然是无法接受的。最近，Chavez 等人 [359] 提出了一个比较有前途的方法。

9.8 文献讨论

关于倒排索引，Zobel 和 Moffat[1798] 给出了一个完整的文献讨论。他们不仅讨论了索引和搜索，而且还有相关的主题。另一个比较好的综述是由 Baeza-Yates 等人 [113] 给出的。[1149] 描述了基于排序的索引构建。[65] 研究了另一个避免排序的可选构建方法。[705] 展示了另一个构建算法。[56，57，58] 讨论了当在倒排索引中处理查询时，如何提前终止的其他技术。压缩倒排索引的进一步结果在 [1738，1739] 中。 394

块寻址倒排索引的想法首先在一个叫做 Glimpse[1078] 的系统中提出来，它也是第一个提出用倒排索引的词汇表来进行复杂模式匹配的系统。[116] 分析了块寻址索引，它提高了性能。

关于压缩的参考文献，可以查看 6.10 节。在 [532] 提出 Elias 编码后，[635] 提出了

Golomb 编码后，而［1709］则给出了如何将关于这些技术运用到压缩的倒排索引的最好阐述。［1158］探索了将这些技术与块寻址和文本压缩结合起来。

后缀树最先由［1678］提出。［1108］给出了线性时间内构建后缀树的方法，但是第一个在线构建方法（从左向右扫描文本）来自［1616］。［108］描述和分析了在后缀树上搜索正则表达式，而容错搜索在［1181，1615］中得到了研究。

［1075］提出了后缀数组和一个构建算法：最坏情况需要 $O(n\log n)$ 次字符比较，平均需要 $O(n\log\log n)$ 次。后缀数组同时被［646］独立发现，它的名字是“PAT 数组”。现在有许多实用的后缀数组构建算法。实际效果最好的可能是一个叫做“deep-shallow”的算法［557］。这篇论文还综述了最相关的可选实用方案。deep-shallow 的免费代码可以从 http://roquefort.di.unipi.it/～ferrax/SuffixArray 下载。在理论方面，有线性时间的构建算法。最简单的只需要数十行的代码［876］，然而它比最佳实用算法要慢，而且需要很多的额外空间。我们已经说明的那个构建大型后缀数组的算法在［647］中提出。最近的一个综述［441］证实这个算法仍然是最佳实用算法之一。对于查询在磁盘上的后缀数组，［1182］展示了一些结果。

关于后缀数组压缩的核心文章都是基于 Ψ 函数［681，1404，1405］和 BWT［556，557］，以及许多后续工作。小波树在［680］中提出，并在［558，1073］中应用于基于 BWT 的索引。有些压缩索引甚至并不是在后缀数组概念上建立的［1177］。最近的一个全面综述是［1184］。那些索引的许多原型在镜像 http://pizzachili.dcc.uchile.cl 和 http://pizzachili.di.unipi.it 上是可以免费获得。

关于签名文件的材料基于［547］。［546］解释了保存签名文件的不同替代方法。

一本关注于序列搜索的实用算法、并深入讨论 9.5 节大部分内容的好书是［1187］。在更加理论的模式匹配方面加以阐述的其他书籍包括［448］。一个公开的、强大的模式匹配工具是 Gnu Grep 工具，它在 http://www.gnu.org 上可下载。

Horspool 算法在［774］中提出，它是原始的 Boyer-Moore 算法［246］的一个简化变种。Boyer-Moore 比 Horspool 取得更长的平移，但是计算它们所需要的时间使得它总体来说更不实用。另一个著名的变种是 Sunday 算法［1545］，但是根据我们的经验，如果加入了“skip-loop”，那么 Horspool 的变种更快［1187］。Knuth-Morris-Pratt 算法［912］因它在最坏情况下也能取得 $O(n)$ 的复杂度而著名，然而在实际中它还是比较慢的。如果保证最坏情况下的线性性能是重要的，那么可以把它与其他算法结合起来，以保持很好的平均情况，同时把坏情况约束在 $O(n)$ 内。Agrep 软件在［1723］中描述，它可以从 glimpse 在
395 ftp://ftp.cs.arizona.edu 上的发布版本中得到。

使用位并行来实现自动机的想法最开始是在［107］（Shift-Or，它是 Shift-And 的一个变种和扩展）和［1724］（Shift-And 和我们已经展示过的近似匹配的扩展）中提出的。后缀自动机和 BDM 在［469］中介绍。它的位并行实现（BNDM）和多个扩展在［1186］中展示。交错 Shift-And 最近在［586］中提出。一个基于 BNDM 构建，并且支持本章涉及的多个模式的软件叫 nrgrep，它在［1176］中描述，并在 http://www.dcc.uchile.cl/～gnavarro/software 上公开。

在图 9-38 中，我们对许多算法进行了实验比较，说明了它们中的哪些在实际中比较好。

正则表达式和将它们转变成 NFA、DFA 和 DFA 最小化的方法，可以在许多经典的书中找到，如［772］。Thompson 算法的原始文献是［1584］。［1188］考虑了其他构建方法，

以及它们在通用 NFA 位并行模拟中的应用。

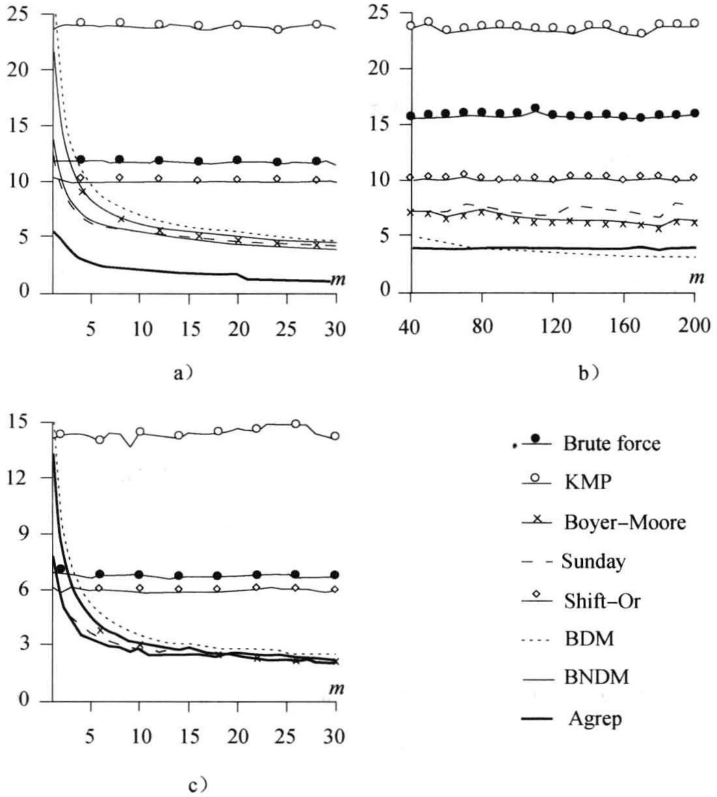


图 9-38　字符串匹配算法的实际比较：a）是在英语文本上的短模式；b）是在 DNA 上的长模式；c）是在随机文本（64 个字母）上的短模式。时间单位是×10 秒/MB（1998 年）

著名的多模式搜索算法有 Aho-Corasick[22]、Commentz-Walter[409]、MultiBDM[447] 和 MultiBOM[1187]。agrep 工具的多模式搜索因它的速度而闻名。

关于近似字符串搜索的一个综述是 [1175]。经典的动态规划方案可以在 [1449] 中找到。我们介绍的将平均时间提高到 $O(kn)$ 的算法来自于 [1614]。一个最坏情况是 $O(kn)$ 396
的算法在 [973] 中描述，但它并不实用。[1614] 使用了 DFA，但是它产生太多的状态。这个问题的位并行方法最先在 [1724] 中开始，尽管现在最快的位并行算法是 [1166] 和 [117]。在所有的筛选算法中，实际最快的算法是基于 [1724] 提出的一个想法，并在 [1180] 中提高。另一个在本章讨论的算法来自 [587]。

关于搜索压缩文本，这里讨论的最复杂的算法在 [1158] 中介绍。在图 9-39 中，以时间-空间复杂度图的形式显示了本章中讨论的大多数技术。

对于空间访问方法的一个全面综述见 [1419]，或者见更早的 [616]。对于 R 树的介绍，可以参见 Guttman[688] 的、有重大影响的文章。在许多后续的变种中，R^* 树 [165]
似乎是性能最好的方法，它使用“强制重新插入”、延迟分割的想法，因此取得了更高的空 397
间利用率，进而得到了更简洁、更短、更快的树。另一个强有力的竞争者是 Hilbert R 树 [865]，它甚至取得了更高的空间利用率，常常比 R^* 树表现得更好。所有这些方法的一个通用框架和实现是 GiST 树 [743]。

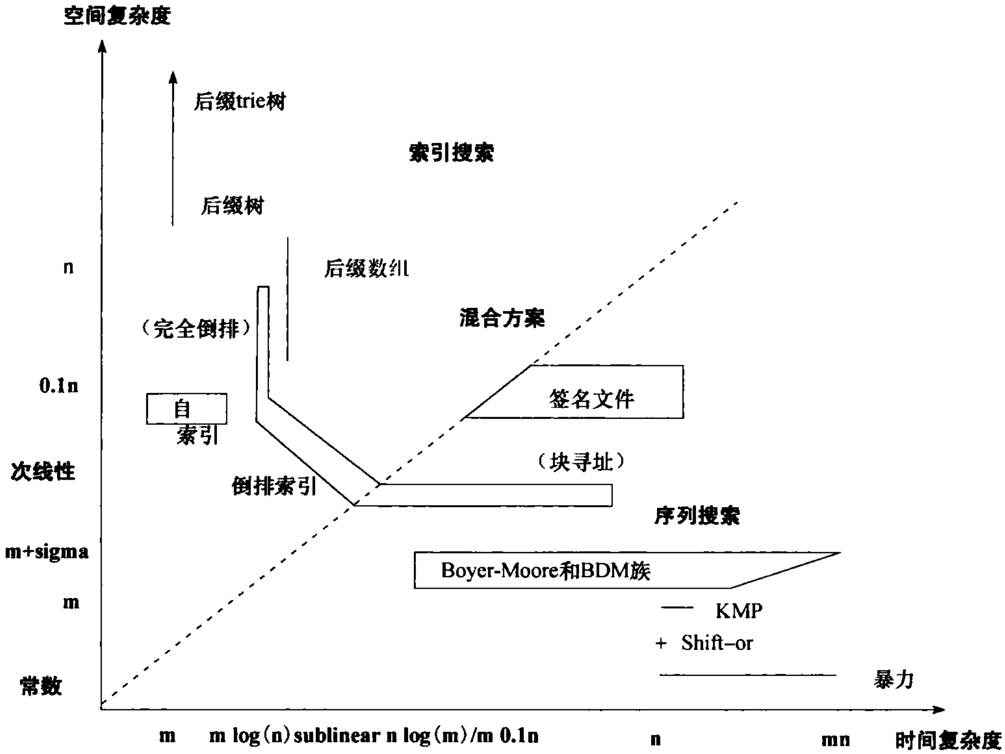


图 9-39 额外空间使用与词搜索时间之间的权衡

对于这些算法，范围搜索在 R 树中是平凡的。最近邻查询需要更仔细的记录保留，以及分支定界算法（如 [1388]）。空间连接（如“找出所有距离在 ε 以内的点对”）也吸引了很多的目光：参见 [264] 中的过滤算法和 [1044] 及 [936] 中的方法。

索引高维地址空间已经吸引了很多的目光。我们首先要提到的数据结构是：TV 树 [1035]，它只使用可用维度中的一部分；SR 树 [879]，它使用超球与矩形作为边界区域；X 树 [185]，它将极高的维度优雅地转变成序列扫描。

对于空间访问方法和选择性估计的分析，“分形维度”的概念已经在它尝试过的每种情况中都给出了十分精确的结果：范围查询 [549]、最近邻查询 [1243]、空间连接 [176]、四叉树 [548]。分形维度的想法是考虑给定点集合的内在维数。例如，考虑在一个 3 维立方体对角线上的点：它的维数是 $E=3$；然而，其内在维数是 $D=1$。使用合适的维数定义，如 Hausdorff 分形维度或者关联分形维度 [1443]，可以发现真实的数据集都有一个分形维数：海岸线是 1.1～1.2，哺乳动物的大脑表面大约是 2.7，雨点的周围大约是 1.3，路段的终点大概是 1.7，不一而足 [549]。

最后，直接在距离函数上操作的访问方法似乎更有前途。正如之前提到的，这些方法只需要一个距离函数，同时它们通常建立一个层次聚类，即一个“超球”的树结构，它包括子超球等。这类方法包括：Burkhard-Keller 方法 [297]、Fixed-query 树 [102]、GNAT 树 [260]、MVP 树 [247] 和 M 树 [385]。这个技术仍然十分年轻：上面的大多数方法是为静态数据集设计的。积极的一面是，它们并不需要进行特征抽取；消极的一面是，它们并不提
398 供可视化和数据挖掘。对这类数据结构的一个很好的综述参见 [360]。

并行与分布式信息检索

——与 Eric Brown 合著

10.1 介绍

如今网络上的电子文本数量多得十分惊人。有人计算过，只是 Web 就包含了超过 200 亿个网页文本，这些文本组成数百 TB 的数据。而且，如第 11 章所讨论的那样，它还在以指数级的速度不断增长，几乎每年就翻一倍。此外，大型的信息服务提供商，如 Lexis-Nexis（见第 16 章），已经积累了超过 TB 大小的数据库。在一个稍微小一点的量级上，最大的企业内网现在也包含数百万个网页。随着硬盘空间变得越来越便宜，以及电子内容变得越来越容易产生、下载和保存，甚至连保存在个人计算机上的私人在线文档集也变得越来越大。

随着文档集不断增长，它们也变得越来越难管理。而且，因为搜索和索引的代价随着文档集的大小不断增长，大型文档集不可避免地会导致更长的响应时间。随着更多的文档加入到系统中，如果没有管理好性能问题，那么操作可能就会恶化，导致系统不可用。相关的问题是，因为搜索引擎的大部分收入来自显示搜索相关的广告，而它的数目又与响应的查询请求数目成正比（见 11.10.1 节），所以搜索引擎提供快速查询处理的能力关系到它的经济生存状况。

为了满足现代搜索环境的苛刻需求，有必要考虑其他架构和算法。本章将探索并行和分布式信息检索技术，它是对第 9 章的补充。并行和分布式计算的应用可以极大地加强传统信息检索算法的可扩展性，支持越来越大的文档集和查询吞吐量，特别是在 Web 上。

因为历史原因，分布式检索最开始是与*联合搜索*（federated search）联系在一起的。也就是说，在多个通常是异质而且互相独立的搜索服务的集合中进行搜索。一个典型的例子是 399
元搜索（metasearch）引擎，它把每个查询发给许多搜索引擎，然后收集不同的结果，合并它们以生成最后的答案。在这种情况下，每个搜索引擎自己维护索引和结果排序。在展示结果之前，元搜索引擎只需要合并它所使用的搜索引擎产生的内容，而不需要知道它们的排序函数。显然，最终的答案依赖于不同搜索引擎所产生的答案。因为其中的一些答案可能需要花较长的时间来产生，所以元搜索引擎通常采用一个计时机制，只考虑那些在给定时间间隔内产生的答案。因此，给用户展示的内容受网络流量状况的影响，这也是元搜索引擎的一个主要缺点。

如果所有的服务器使用同样的软件，那么就产生了另一类型的分布式信息检索系统。在这种情况下，所有服务器的排序函数都是已知的，这意味着可以合作生成一个更一致、更高质量的排序。尽管实际上搜索服务分布在不同物理位置的许多机器上，但这些服务器是同质的，并且在中央控制下操作，于是就可能给用户只有一个搜索系统的错觉。因为这和 Web 的情况类似，我们叫它*分布式检索*。

在分布式检索系统中，每个服务中心可能是由几千个合作提供服务的服务器组成。例如，考虑一个在大小为 N 的文本集上、每秒能回答 N_{qps} 个查询的检索系统，它使用一个只有单个处理器的服务器。为了增加查询吞吐量和文档集大小，可以增加更多的机器来形成一个由快速局域网连接的服务器集群 C。于是我们就有了一个*基于集群的搜索系统*。通过将索引划分到集群中的服务器上，可以显著地增大文档集。这就是*索引划分*（index partitio-

ning）的问题。给定一个已经划分了索引的机器集群 C，我们同样希望最大化每秒处理的查询数目 N_{qps}，以降低总体开销。这就是*查询处理*（query processing）问题，它受到索引划分的影响。

一旦我们有了高效的机器集群，我们可能仍然需要提高查询吞吐量来满足不断增长的需求。这通常通过复制整个集群来实现。一个配置集群的常见方法是把完整的索引在每个集群上都复制一遍。在这种情况下，每个集群都能独立处理任何查询。这是现代搜索引擎的配置，即一个由多个集群组成的分布式检索系统。这个架构能不断地提高查询吞吐量，同时通过使服务器靠近用户来降低时延。

如果完整的索引无法保存在单个集群上，那么我们可以将索引划分到多个集群上。由此能够产生许多关于可扩展性的解决方案，如图 10-1 所示。为了组织这些方案，在纵轴上显示复制，而在横轴上显示划分。基于集群的搜索引擎组织方式是 Web 检索的关键，它的应用将在 11.4.2 节讨论。

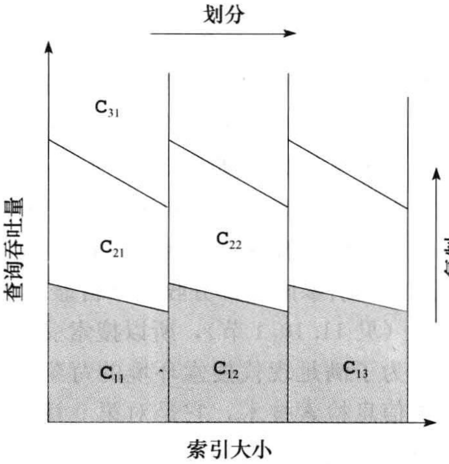


图 10-1 大型信息检索系统中的可扩展性问题，其中每个 C_{ij} 代表一个计算机集群（改编自［1153］）。每个集群的性能的梯形形状接近这样的事实：当集群的硬件资源并未饱和时，查询吞吐量应该更大

第四个提供可扩展性的选择是对服务器增加更多的处理器、内存和磁盘，以增加并行处理能力，不过目前使用得并不多。在这种情况下，处理器通过内部总线紧密地连接在一起，这使得传输数据的带宽比局域网更高。而且，机器可以实现一个内存共享的架构，其中多个处理器可以并发地访问同样
400 的内存。我们称之为*并行信息检索系统*，将在 10.4 节详细讨论。尽管最近并行信息检索系统因为开销考虑并未被采用，但是讨论它们还是很重要的，因为它们体现了很多对于完整理解分布式信息检索十分重要的概念。例如，集群中的服务器可能有多于一个的处理器。

第五个非常独特的分布式架构是由对等网络（Peer-to-Peer Network，P2P）提供的，其中大量自治的计算机分布在各地。每一个计算机运行相同的通信协议，并且用该协议实现相似的能力和任务。在这种情况下，每个端点（peer）建立它自己的索引版本，并能解决本地查询。P2P 系统比客户端/服务器架构要动态得多，它允许计算机随意加入或者离开网络。因此，P2P 网络通常比联合搜索系统有多得多的参与结点（计算机），我们将在 10.7 节详细讨论。

本章的接下来部分安排如下。10.2 节展示已经讨论过的、不同类型的并行和分布式信息检索系统的分类。10.3 节涉及如何划分文档集和索引。10.4 节探索在并行平台上实现信息检索的技术。10.5 节讨论基于集群的信息检索，这是现代搜索引擎所选择的架构。10.6 节的注意力转向分布式信息检索，包括索引和查询处理。10.7 节将讨论联合搜索，10.8 节
401 将讨论对等网络。最后，讨论未来的趋势，然后以文献讨论来总结。

10.2 分布式信息检索系统的分类

因为并行和分布式信息检索系统可以用许多不同的方式组织，所以我们提出一种宽泛的分类方法，它足够通用，包含主要的系统和它们的变种。如图 10-2 所示，它基于三个主要

的特征，我们用它们来描绘各种类型的并行和分布式系统。

	单处理器	多处理器	
	相同软件	相同软件	不同软件
内部通信	标准搜索	并行搜索	并行搜索
局域网通信	n/a	基于集群的搜索	本地联合搜索
互联网通信	n/a	分布式搜索（P2P）	联合搜索（P2P）

图 10-2　分布式和并行信息检索系统的分类，根据处理器个数、使用的软件个数和通信媒介组织 402

- **处理器个数**——如果只有一个处理器，那么我们当然只有一台机器。这种情况下的通信是在系统内部，并通过处理器-内存总线来完成。如果使用多个处理器，那么处理器可能是单台并行机的一部分，或者是多个独立机器的一部分。在第一种情况中，处理器间的通信是通过内部总线完成。在第二种情况中，处理器间的通信是通过局域网或互联网来完成。
- **相同或不同的软件**——这里的问题是信息检索系统中的服务器是使用相同的软件，还是使用不同的软件。在并行和基于集群的机器的情况下，相同的软件相对容易部署。当服务器完全不同，或属于不同的组织机构时，就常常会出现不同软件的情况。在这种情况下，引擎常常是一个元搜索引擎，它收集由不同服务器产生的结果，并把它们合并到单个结果列表中。也有可能因为有历史遗留下来的应用，同样的组织机构需要在一个或多个服务器上运行不同的软件。
- **通信媒介**——这里的问题是处理器间通信，为此，基本上有三种可以使用的技术：内部总线、局域网和互联网。在内部总线的情况中，通信协议是轻量级的，有效的传输速率可以达到每秒 GB 级或更多。在局域网的情况中，使用了一个重量级的通信协议，有效的传输速率是在每秒 MB 级的范围。在更大的互联网的情况中，可能需要在协议层加上额外的控制，如拥塞控制，有效的传输速率通常在每秒 KB 级范围。尽管网络技术在不断地革新，这些范围很有可能会变化，但是我们应该认识到，局域网和互联网的协议以及通信时延将继续变得愈发严重，这将导致这三种情况的传输速率达到一个数量级或更大的差别。

如图 10-2 所展示的分类区分了七种类型的分布式和并行系统，包括：

- **标准搜索**——这个引擎在包含一个处理器的单机上实现。这是最简单的形式，通常被学生和研究人员所采用，以试验新的排序函数、新的索引器、新的爬虫和新的界面。它的主要好处是快速部署以及低维护代价。对于生产系统，这个形式只适用于手头的文档集较小的情况。
- **并行搜索（SIMD）**——这个引擎在由多个处理器组成的并行机上实现。这些处理器可能有它们本地的内存，但较常见的是它们有共享的内存（带有一个互斥总线控制机制来避免竞争条件）。所有的处理器执行同样的软件，这里所提到的 SIMD 表示单指令流、多数据流（single instruction stream，multiple data streams），将在 10.4 节讨论。
- **并行搜索（MIMD）**——这个引擎在由多个处理器组成的并行机上实现。这些处理器都有本地内存，并通过一个十分快速的内部交换结构（传输速率和总线一样）来通信。处理器可能执行不同的软件，这里提到的 MIMD 代表多指令流、多数据流

(multiple instruction streams, multiple data streams), 将在 10.4 节讨论。

- **基于集群的搜索**——这个引擎在一个由多台通过局域网互联的机器所组成的集群上实现。利用各种各样的机器将海量文档集分割成各种各样的本地子文档集，便于进行并发搜索。这是今天的商业搜索引擎更青睐的形式。
- **本地联合搜索**——这个引擎把每个查询发送到运行不同软件的多个独立服务器上。这些软件可以很不一样，如传统的布尔搜索系统、基于向量的系统和现代的 Web 搜索系统。不同系统产生的结果被该引擎收集，并以启发式的方式合并，然后以单个列表的形式返回给用户。本地联合搜索尽管很少使用，但还是会在一些大型组织中使用。它们包含许多历史遗留的系统，而没有收集和维护合作信息的中央进程。
- **分布式搜索**——这个引擎在分布在互联网上的多个处理器上运行，但都使用相同软件。这对通常使用的基于集群的形式来说是个有趣的替代方案，它最近已经吸引了
403 越来越多的研究活动，我们下面会进行具体讨论。
- **联合搜索**——通常的情况是元搜索引擎把每个查询发送给位于互联网中不同地方的多个服务器上，然后收集结果，启发式地合并它们，最后将单个结果列表展示给用户。

根据中央化和控制的程度，分布式信息检索系统也可以按照如下两个基本的网络协议类型来区分：

- **客户端/服务器**：网络中的机器有两个基本的类型：服务器，或者客户端。服务器是服务的提供者，客户端是消费者。在分布式信息检索的情况中，服务基本上就是搜索，客户端机器产生查询（或服务请求），然后发送到服务器。服务器处理这些查询（或请求），生成答案，并把它们发回客户端来结束该服务。使用客户端/服务器协议的分布式信息检索系统有三种基本类型：基于集群的搜索、联合搜索和分布式搜索。在联合搜索中，网络上的服务器可能使用不同的软件，只有连接服务器和客户端的协议需要标准化。在分布式的和基于集群的搜索中，所有的服务器都使用同样的软件（在并行搜索中也如此）。联合搜索和分布式搜索系统在互联网上进行操作，而基于集群的搜索系统通过局域网更紧密地连接。图 10-2 所示的所有七种信息检索系统的组织都可在客户端/服务器的模式下工作。而且，如果把浏览器看成客户端的话，那么所有的 Web 搜索引擎都以客户端/服务器模式工作。
- **对等（P2P）**：网络中的每台机器都有同等的能力和责任。也就是说，每台机器都能作为客户端或者服务器运行。尽管不同的机器可能使用不同的软件，但它们必须通过一种通用的 P2P 协议来通信。因为这个原因，最常见的情况是所有的端点都使用相同的软件。P2P 网络要动态得多，因为机器可以在任何时候加入或者离开网络，我们稍后会对其讨论。图 10-2 的系统组织中，有两种类型可以在 P2P 模式下运行：分布式搜索和联合搜索（用一个 P2P 标签来标记）。

关于中央化的水平，我们可以说，在分布式搜索系统中控制是完全的，在联合搜索中是部分的，而在 P2P 信息检索系统中根本就不存在中央控制。

10.3 数据划分

因为信息检索计算任务通常可以刻画成“把少量的处理过程（对于每份数据）应用到大量的数据上”，所以如何将计算划分到不同的机器上就是如何划分数据的问题，如怎样划分文档集和索引。我们这里讨论的数据划分技术可以应用到并行和分布式信息检索系统。

图 10-3 以一个高层次的视角展示了在向量空间模型（见 3.2.6 节）中常用的搜索算法处理数据的过程。每一行表示一篇文档 d_j，每一列表示一个索引项 k_i。这里，k_i 可能是单个词、短语、概念或者更抽象的索引项，如 LSI 向量中的一个维度或者文档签名中的一位。矩阵中的元素 $w_{i,j}$ 是项-文档权重，详见 3.2.3 节。与某个特定文档相关的所有项权重组成一个带权重的文档向量 $\vec{d}_j$： 404

$$\vec{d}_j = (w_{1,j}, \cdots, w_{t,j})$$

	索引项					
文档	k_1	k_2	…	k_i	…	k_t
d_1	$w_{1,1}$	$w_{2,1}$	…	$w_{i,1}$	…	$w_{t,1}$
d_2	$w_{1,2}$	$w_{2,2}$	…	$w_{i,2}$	…	$w_{t,2}$
…	…	…	…	…	…	…
d_j	$w_{1,j}$	$w_{2,j}$	…	$w_{i,j}$	…	$w_{t,j}$
…	…	…	…	…	…	…
d_N	$w_{1,N}$	$w_{2,N}$	…	$w_{i,N}$	…	$w_{t,N}$

图 10-3 搜索算法处理的基本数据元素

在搜索过程中，查询也用索引项的权重向量来表示，$\vec{q} = (w_{1,q}, \cdots, w_{t,q})$，而搜索算法通过应用匹配函数 $F(\vec{d}_j, \vec{q}) = \text{sim}(d_j, q)$来给每篇文档打分，第 3 章讨论了许多这样的算法。

对于划分数据，这个高层次的数据视角揭示了两个可能的数据划分方法。第一个方法是**文档划分**（document partition），它水平地切分数据矩阵，将文档分在多个子任务中。文档集中的 N 篇文档分布在系统中的 P 个处理器上，生成 P 个子文档集，每个估计有 N/P 篇文档。在查询过程中，每个并行过程（每个处理器一个）在分配给它的 N/P 篇文档的子文档集上执行该查询；最后，来自每个子文档集的结果被合并到一个最终的结果列表中。第二个方法是**项划分**（term partitioning），它垂直地切分数据矩阵，将索引项分到 P 个处理器上，使得对每篇文档的执行过程都传播到系统中的多个处理器上。现在我们考虑将这两个划分方案用于主索引结构。

10.3.1 文档集划分

在并行或分布式信息检索系统中，用于将文档分配给搜索处理器或服务器的过程依赖于多个因素。首先，我们必须考虑系统是否是中央管理的，如在并行信息检索系统中常常是这样的情况。在这种情况下，生成划分的最简单方法是随机选择文档，同时每个划分大概都包含 N/P 篇文档。一个更结构化的方法是使用 k 均值聚类算法来根据主题对文档集划分[978, 1041]，见 8.3.1 节。其他可能的划分策略包括，根据语言或者其他内在的数据特征来划分，如地理位置等。

在由独立管理的异质搜索服务器组成的分布式系统中，文档集是独立建立和维护的。在 405
这种情况下，没有中央节点来控制文档的划分过程，于是怎样划分文档的问题也是无意义的。然而，有可能每个独立的搜索服务器都关注于一个特定的主题领域，结果就是将文档按照语义划分成关注于特定主题领域的多个分布式的文档集。这种情况在元搜索引擎中很常见，它提供对各种各样的后端搜索服务提供者的集中化的访问。另一个可能是，每个服务器负责一种不同的语言或者地理区域。

如果是中央化管理的分布式系统，那么就会有更多的选择。第一个选择是在所有的搜索服务器上简单地复制整个文档集。当文档集足够小且能够保存在单个搜索服务器上，但需要很高的可用性和查询吞吐量时，这是合理的。在这个场景中，利用多任务来实现系统中的并行化（见图 10-5），而代理程序的任务是将查询分配到搜索服务器上，并平衡在服务器上的负载。

第二个选择是文档随机分布。当出于性能原因必须把一个大型文档集分散存储时，这种选择是比较合适的。但是，这些文档将总是被看做单个逻辑文档集的一部分进行搜索。代理

程序把每个查询广播到所有服务器，然后组合所有的结果给用户。

最后一个选择是显式的语义文档划分。这里，文档要么已经根据技术准则等组织成语义上有意义的文档集，要么使用自动的聚类或分类过程来将文档划分成与主题相关的文档集。这是最令人感兴趣，也是最复杂的情况，我们下面会展开讨论。

有一些论文讨论了如何划分一个文档集，使得每个文档集都能很好地和其他文档集分开。在这个语境下，“很好地分开”的意思是，在这种划分下，每个查询都能映射到某个包含最多相关文档的文档集中。为了构建这样一种映射，可以使用查询日志。Puppin 等人 [1304] 使用查询日志，并用带有所有返回文档答案的查询来表示文档。这样的表示能够用一种共聚类算法来建立查询聚类和文档聚类。然后利用共聚类的结果，基于文档聚类来划分文档集，并基于查询聚类和文档聚类建立文档集选择函数。它们的结果显示，使用这种技术，其性能能够超过目前已知对于文本数据最好的文档集选择函数 CORI[320]（见 10.3.2 节）。CORI 只是基于在文档集中的信息，而 Puppin 等人的技术的优势是根据从用户使用模式的信息构建的模型进行划分，这对于未来的查询可能也是有效的。另一个有意思的结果是，查询日志不仅能有效地划分文档集，而且可以识别出未来查询不可能检出的某个文档子集。在这篇文章中，他们展示了对于一个特定的大型文档集，这个子集大概占到所有文档的 53%。

然而，设计一个文档集划分算法，使得减少执行查询时涉及的服务器个数，以及平衡总
406 体的查询负载，仍然是一个开放的问题。

10.3.2 文档集选择

在许多情况下，特别是在分布式和联合搜索中（见 10.6 节和 10.7 节），文档集是预先决定的，无法改变。在那种情况下，文档集选择也称为源选择或查询路由，是决定哪个文档集最有可能包含与当前查询相关的文档，因此应该接收查询来进行处理的过程。一个方法是，总是假设每个文档集有同等的可能性包含相关的文档，故简单地将查询广播到所有文档集。当文档是随机划分的，或者文档集之间有很强的语义重叠时，这个方法是合适的。

当文档集被划分成语义上有意义的子文档集，或者每次搜索全部文档集的代价十分高时，可以根据文档集包含相关文档的可能性对文档集进行排序。基本的技术是把每个文档集看做一个大的文档，对这些文档集建立索引，并对每个文档集执行查询，生成一个排序的文档集列表。排序可以基于 3.2.6 节讨论的向量模型。为此，我们需要计算文档集 C_j 的项权重，它可以按照如下过程完成。令 $w_{c,ij}$ 表示文档集 C_j 中的项 k_i 的权重，那么，

$$w_{c,ij} = f_{c,ij} \times IDF_{c,i}$$

其中 $f_{c,ij}$ 是项 k_i 在所有文档集 C_j 中的总出现频率，而 $IDF_{c,i}$ 是反比文档集频率，即

$$IDF_{c,i} = \log\left(\frac{N_c}{n_{c,i}}\right)$$

其中，N_c 是文档集的数目，$n_{c,i}$ 是出现项 k_i 的文档集的个数。这些权重用于生成查询和文档集向量，它们的相似度用向量模型的余弦相似度公式计算，见 3.2.6 节。

这个方法的一个问题是，尽管某个文档集可能得到一个较高的查询相关度分数，但是有可能在该文档集中没有任何单个文档有较高的查询相关度分数（这种情况叫做误检（false drop））。Moffat 和 Zobel[1152] 提出通过以一系列文档块的形式对每个文档集建立索引，每个块包含 B 篇文档，来避免这个问题。当 B 等于 1 时，这就等同于索引所有文档，将之当做单个文档集。当 B 等于在每个文档集中的文档数目时，这就等同于原来的解决方法。

通过变化 B，就可以在文档集的索引大小和误检的可能性之间进行权衡。

Voorhees[1652] 提出了搜索文档集索引的另一个选择，它同时利用训练查询来对分布式的文档集建立一个内容模型。当一个新的查询提交到系统时，计算出它和训练查询的相似度，然后内容模型就用于决定应该搜索哪个文档集，以及从每个文档集应该返回多少文档数。

广义 GIOSS 系统 [666] 根据文档集中包含查询项的文档数目和某个项在所有文档中的总权重对文档集排序。更具体地说，它需要两个向量来估计排序：项在每个文档集中的文档 407
频率和每个项在文档集的所有文档中的权重总和。基于这些信息，作者提出了两个估计算子来预测文档集中与查询的相似度大于阈值 l 的文档个数。Max(l) 估计算子假设在文档库中查询项以最高程度共现。另一方面，Sum(l) 估计算子假设查询项在任何文档中都不一起出现。然后，相似度就用标准的余弦函数计算。

CORI 系统 [323] 将文档集当做文档来排序，使用 Inquery[322] 推理网方法来检索文档（见 3.5.4 节）。为了对文档集进行排序，每个文档集用它的索引项表示，其中项的权重由文档频率和文档集频率这些统计数据决定。查询获得的文档集通过它们的排名分数进行聚类，并选择前 k 个文档集。

GIOSS 和 CORI 这两个方法在选择最好的数据源来回答查询时，都未把搜索结果的质量考虑进去。因此，对于给定的查询，具有相似的查询项统计数据的文档集有同等的重要性。然而，因为它们可能有不同的排序函数，所以结果的质量也可能是不一样的（见 7.2.6 节）。

其他一些早期的方法对文档集的词汇表进行索引，根据文档集中的项频和项的文档频率的相似度分数将文档集当做伪文档来排序 [1797]。Yuwono 和 Lee[1761] 的工作尝试确定文档集中的哪些项将该文档集与其他文档集区分开来，并给更具判别能力的项更高的权重。

D'Souza 等人 [513] 调查了另一类不同的文档集选择方法，它们使用代理文档而非只来自于不同文档集的索引项信息。他们描述了文档排序方法，其中代理文档是通过文档中前 n 个或者最好的 n 个索引项产生。这使得能够使用文档的相似度分数来对文档集排序。

Fuhr[600] 介绍了一个文档集选择的决策理论框架（Decision Theoretic Framework, DTF），对每个文档集，根据一个代价函数估计最优的检出文档数。这个代价函数依赖于通信代价和获取文档的代价，无论文档是相关的还是不相关的。Nottelmann 和 Fuhr 还介绍了在 DTF 中，根据检出文档的质量估计代价的方法 [1210]，以及如何使用 CORI 来估计在文档集中相关文档的个数 [1211]。

后来，Si 等人 [1475] 提出了一个用于文档集选择的语言模型框架。GIOSS 系统假设排序分数在文档集间是可比较的，而 Si 等人提出的系统假设这个分数可能是无法比较的。在这种情况下，文档集通过查询采样来描述，其中的查询是从某个背景分布采集的单个项。对于每个查询，根据前 4 篇文档创建语言模型。他们发现，一个文档集只用 300 篇文档就足够表示了。文档集像用语言模型方法检索文档那样进行排序，其中被检索的“文档”是文档集的形式（见 3.5.2 节）。

仍然还有一些重要的悬而未决的问题。例如，文档集选择和文档集划分之间的相互依赖并未很好地得到研究。特别地，当使用某个特定的划分技术时，一个好的文档集选择机制可 408
能会产生不平衡的查询负载。

10.3.3 倒排索引划分

我们先讨论使用文档划分进行倒排索引划分的方法，然后我们讨论项划分。在这两种情

况中，我们都描述索引和基本的查询处理步骤。

在使用倒排索引的系统中有两个文档划分的方法，即逻辑文档划分和物理文档划分。第一个适合于具有共享内存的并行系统；而后者是分布式系统的唯一选择。

1. 逻辑文档划分

在这种情况中，数据划分在逻辑上使用和原有序列算法（见 9.2.5 节）一样的倒排索引。该倒排索引经过扩展，使得每个并行的进程（一个处理器一个进程）能够直接访问与处理器的子文档集相关的那部分索引。每个词汇表条目经过扩展，包含指向对应的倒排索引的 P 个指针，其中第 j 个指针是与第 j 个处理器中的子文档集相关的倒排索引中的文档块的索引。如图 10-4 所示，其中项 i 的词汇表条目包含四个指向项 i 的倒排列表的指针，每个并行进程一个指针（$P=4$）。

当查询提交到系统时，代理程序（见图 10-6）先保证必需的词汇表和倒排索引条目装入共享内存，其中所有的并行进程能够访问这个共享的副本。然后，代理程序初始化 P 个并行进程来执行查询。每个进程使用扩展的词汇表来访问倒排索引中的适当条目，在它的子文档集上执行相同的文档打分算法。因为在查询处理过程中所有的索引操作都是只读的，所以在访问共享词汇表和倒排索引的进程间没有锁竞争的问题。搜索进程在一个共享的文档分数累加器数组中记录文档的分数，并在结束时通知代理程序。对累加器的更新也没有锁冲突的问题，因为不同搜索进程进行打分的子文档集是互斥的。在每个搜索进程结束后，代理程序对文档分数累加器数组进行排序，并产生最后的排序文档列表。

在构建倒排索引时，对逻辑划分的文档进行索引的进程可以使用 Brown[282] 描述的一种索引方案（见 9.2.5 节），以利用并行处理器。首先，索引器将文档在处理器间划分。其次，它分配文档标识符，使得在划分 i 中的所有标识符都小于在划分 $i+1$ 中的所有标识符。然后，索引器以并行的方式在每个处理器上运行独立的索引进程。每个索引进程都生成一批按照索引项排序的倒排索引。在所有索引都产生后，执行合并步骤来产生最后的倒排索引。因为每个倒排索引都以同样的方式排序，所以使用一个基于二叉堆的优先队列来合并对应于当前索引项的所有部分倒排索引。这些部分索引按照划分编号的顺序连接起来，产生最后的倒排列表；然后生成索引项对应的词汇表条目，它包含额外的索引
409 指针，如图 10-4 所示。

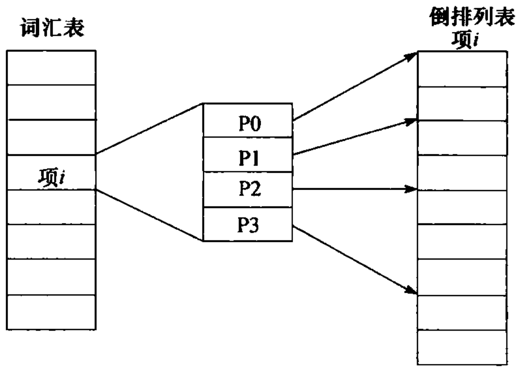


图 10-4 用于文档划分的扩展词汇表条目

2. 物理文档划分

在文档划分的第二种方法中，文档被物理地划分成分离的、自包含的子文档集，每个并行处理器或分布式服务器有一个子文档集。每个子文档集有自己的倒排索引，在执行查询时搜索进程间并不共享任何东西。当查询提交到系统中时，代理程序将查询分发给所有的并行搜索进程。每个并行的搜索进程在自己的部分文档集上进行查询，产生一个本地的、中间命中列表。然后代理程序从所有并行的搜索进程收集中间命中列表，把它们合并成一个最终的命中列表。

使用一个基于二叉堆的优先队列 [425]，可以将 P 个中间命中列表高效地合并起来。n 个元素的优先队列有这样一个性质，元素 i 大于元素 $2i$ 和 $2i+1$，其中 i 的范围是 $1\sim n$。优先队列并不是完全排序的，但是最大的元素总是能马上得到（即在 $O(1)$ 时间内），并能在

$O(\log n)$ 时间内抽取出来。把一个元素插入到优先队列中同样可以在 $O(\log n)$ 时间内完成。为了合并中间命中列表，需要创建一个 P 个元素的优先队列，即把来自每个中间命中列表的第一个条目都插入到队列中，所需的创建时间是 $O(P\log P)$。

为了生成由排名前 k 位（top$k$）的文档组成的最终（全局）命中列表，从优先队列中抽取 k 个元素。每次在从优先队列中抽取一个元素之后，都从该元素所在的中间命中列表中取出一个新元素插入到优先队列中。P 个中间命中列表可以在 $O((P+k)\log P)$ 时间内被合并成一个 k 个元素的最终命中列表。 410

上面描述的合并步骤假设并行搜索进程产生的文档分数是全局一致的，可以直接合并。根据所使用的排序算法，每个并行搜索进程可能需要全局的索引项统计数据，以便产生全局一致的文档分数。有两个基本的方法来收集全局的索引项统计信息。第一个方法是在索引时计算全局的项统计数据，并将这些数据和每个子文档集一起保存。第二个方法是每个查询处理分两个步骤进行。在第一步中，代理程序从每个搜索进程中收集子文档集的索引项统计数据，然后把它们合并成全局的项统计数据。在第二步中，代理程序将查询和全局项统计数据分发给搜索进程，查询执行过程则和之前一样。第一个方法提供了更好的查询处理性能，代价是更加复杂的索引；而第二个方法能够独立地建立和维护子文档集，代价是在执行查询过程中有双倍的通信代价。

为了构建用于物理划分的文档集倒排索引，每个处理器或服务器并行地创建和子文档集对应的完整索引。如果全局文档集统计数据保存在单独的词汇表中，那么必须执行一个合并操作来从所有划分中累加得到全局统计数据，并分发到每个划分的词汇表中。

在复制文档集的情况下，有两种方法来处理索引文档。在第一种方法中，每个搜索服务器单独地索引文档的副本。在第二种方法中，对每个服务器分配一个互斥的子文档集来进行索引，这些索引子集被复制到所有搜索服务器上。在每个搜索服务器上需要对这些子集进行合并来创建最终的索引（可以使用我们之前提到的方法)。在任何一种方法中，文档更新和删除都必须广播到所有的服务器上。文档加入可以立即广播到系统中，或者根据加入的频率以及系统所必须体现的更新速度，对它们进行批处理和划分。

3. 比较

逻辑文档划分比具有相似并行性的物理文档划分需要更少的通信，所以它有可能提供更好的总体性能。另一方面，物理文档划分提供更多的灵活性（如可以单独搜索文档划分)。并且，如果要将已有的信息检索系统转换成并行系统的话，使用物理文档划分更加简单，而对于分布式信息检索系统，这是唯一可行的方案。无论对于哪种文档划分方案，线程为创建搜索进程、控制它们的操作和它们之间的通信提供了一个方便的编程模式。线程在一些现代的编程语言（如 Java[991]）中是原生支持的，在其他一些语言中也是以标准的方式很好地支持（如在 C 或 C++中的 POSIX 线程)。线程包使得编程人员能够使用高层次抽象化的并发执行、通信和同步来开发并行程序。然后编译器和运行时系统将这些抽象化映射到高效的操作系统服务和共享内存操作上。 411

4. 项划分

当在基于倒排索引的系统中使用项划分时，对文档集生成单个倒排索引（使用上面描述的用于逻辑文档划分的并行构建技术)，然后将倒排索引划分到多个处理器上。在执行查询过程中，查询被分解成索引项，并且每个索引项被分发到包含对应倒排索引的处理器上。这些处理器创建带有部分文档分数的命中列表，并将它们返回给代理程序。然后代理程序根据查询的语义将命中列表合并起来。对于布尔查询，对命中列表进行适当的合并、求交集或求

差集的操作。对于排序的自由文本查询，包含项分数的命中列表必须根据排序公式的语义来合并。

另一方面，项划分允许并行地处理查询，因为每个处理器能够回答不同的部分查询。然而，查询负载并不一定是均衡的 [85]，于是并发的部分好处就丢失了。因此，主要的任务是划分索引，使得：

- 相互联系的处理器或服务器的数目最小。
- 负载平均地分布到所有可用的处理器或服务器上。

5. 总体比较

相比之下，文档划分能提供比项划分更加简单的倒排索引构建和维护。Jeong 和 Omiecinski[833] 指出，它们在查询过程中相对的性能依赖于项的分布。假设每个处理器有它自己的 I/O 通道和硬盘，当项在文档和查询中的分布非常不均衡时，文档划分的性能更好。当项在用户查询中均匀分布时（这个条件和自然文本更近似），项划分的性能更好。例如，使用 TREC 数据，Ribeiro-Neto 和 Barbosa[145，1349] 指出，对于长查询，项划分可能要快两倍，对十分短的查询（如 Web）要快 5～10 倍。然而，其他作者已经发现了相反的结果。

事实上，Webber 等人 [1673] 说明了项划分能够导致更低的资源使用率。更具体来说，它极大地减少了硬盘访问次数和数据交换量。尽管文档划分仍然在吞吐量方便表现得更好，但他们显示，项划分有可能取得更高的值。

最近，Martin 等人 [255，1087，1088，1089，1090] 提出了一个新的架构，它结合了异步操作、同步操作以及一个时间片轮转技术来处理查询。这个技术能够以相同的方式应用到文档和项划分，因而可以说这个技术提供了一个公平的比较环境。在这种情况下，项划分表现得更好。

虽然文档划分方法保证了负载均衡，每个查询的代价被平等地分配出去，但文档划分系统主要的缺点就是，执行了很多不需要的操作来查询那些可能只包含很少（甚至没有）相关
412 文档的子文档集。

项划分的主要缺点是必须建立和维护整个全局索引，这限制了它的可扩展性。因此，它在实际的大规模搜索引擎中并不实用（见第 11 章）。此外，项划分的响应时间有更大的变数，要解决这个问题需要更加复杂的平衡机制。

然而，如果以某种混合的方案和文档划分结合起来，那么项划分在分布式信息检索系统中仍然可能是有用的。因为这个原因，对这种系统来说，有一种好的划分方法仍然是需要的。

10.3.4 划分其他索引

1. 后缀数组

我们可以直接将文档划分应用到后缀数组中。正如对于倒排索引的物理文档划分，文档集被划分到 P 个处理器，每个划分被当做独立、自包含的文档集。然后这个系统可以将 9.4.4 节描述的后缀数组构建技术应用到每个划分上，改进之处在于所有的划分可以并发地进行索引。在搜索过程中，代理程序将查询广播给所有的搜索进程，收集中间结果，然后将中间结果合并到一个最终的命中列表。

如果所有的文档都保存在同一个文档集中，那么我们仍然可以利用并行处理器来减少索引时间。大文本的后缀数组构建算法（见 9.4.4 节）的一个有趣的性质是所有对部分索引的合并过程都是独立的。因此，所有的 $O((n/M)^2)$ 次合并操作都可以在单独的处理器上并行

地运行。在所有的合并完成后，对每个部分索引的计数必须收集起来，并执行最后的索引合并。

后缀数组的项划分是把单个后缀数组分发到多个处理器上，这样每个处理器负责数组的某个字典序区间。在查询处理过程中，代理程序将查询分发给包含后缀数组相关部分的处理器，并合并结果。注意，当搜索后缀数组时，所有的处理器都需要访问整个文本。在拥有共享内存的单个并行计算机（如 SMP 系统）上，这不是问题，因为文本可以缓存在共享内存中。然而，如果共享内存并不可用，并且通信代价很高，如分布式系统（工作站网络等），那么这也可能是个问题。

2. 签名文件

为了在使用签名文件的系统中实现文档划分，文档像之前一样被划分到多个处理器上，每个处理器对它的文档划分生成签名。在查询时，代理程序为查询生成签名，然后把它分发到所有并行的进程中。每个进程在本地执行查询签名，好像它的文档划分是单独、自包含的文档集。然后结果被发送到代理程序，代理程序再把它们合并到一个最终的命中列表给用户。对于布尔查询，最后的结果只是每个处理器返回结果的并集。对于排序的查询，按照上 413
面所说的用于倒排索引的实现方法对排序的命中列表进行合并。

为了在基于签名文件的系统中应用项划分技术，不得不使用按位切分的签名文件[1239]，并将位切片划分给处理器。然而，合并来自每个处理器的中间结果并消除误检，所需要进行的串行操作工作量严重限制了这种组织方式所提供的加速比 S。于是，这种组织方式并不受推荐。

10.4 并行信息检索

10.4.1 介绍

我们可以从两个方向开发并行信息检索算法。一个可能的方案是开发新的检索策略，直接实现并行。例如，文本搜索过程可以建立在神经网络上。神经网络（见 3.4.3 节）根据人脑建模，用大量的结点（神经元）来解决问题，每个结点有一些输入、一个阈值和一个输出。一个结点的输出与一个或多个其他结点的输入相连接，并在这个网络的边界上定义好系统最初的输入和最终的输出。一个结点的输出值由结点的输入和阈值的权重函数决定。通过训练过程来学习网络中权重和阈值的合理值。计算的过程是，把输入值应用到网络中，计算每个活跃结点的输出值，并调整网络中的这些权重与阈值，直到得到最后的输出值。神经网络天然地能够在 SIMD 硬件上并行实现（SIMD 的定义见下文）。这个方法的挑战是如何以这样的方式定义检索任务，即如何很好地映射到计算范式。

另一个可能的方案是将已有的、研究较多的信息检索算法调整成并行过程。这是我们在本章接下来的部分所要考虑的方法。将已有算法调整成并行实现所需的修改依赖于目标并行平台。我们将研究把某些检索算法应用到 MIMD 和 SIMD 架构（它们的定义见下文）的技术。因为并行信息检索仍然是一个十分活跃的研究领域，只有很少技术已经脱颖而出成为标准技术。因此，我们只是展现一些已经完成的工作，而不是比较它们的优劣。

1. 并行计算

并行计算能并发地应用多个处理器解决单个问题，其中每个处理器用来解决问题的某个部分。有了并行计算，解决问题所需的总时间就可以减少到最长运行部分所需的时间。只要问题可以进一步分解成更多能够并发运行的部分，就可以加入更多的处理器到系统中，减少

414 解决问题所需的时间，并可以扩展到更大的问题上去。

处理器可以按各种方式组合来形成分布式架构。Flynn[570] 基于架构中指令和数据流的数量定义了常用的并行架构分类体系。这个分类体系包括四类：

- SISD：单指令流，单数据流。
- SIMD：单指令流，多数据流。
- MISD：多指令流，单数据流。
- MIMD：多指令流，多数据流。

SISD 类包括运行串行程序的传统的冯·诺依曼 [300] 计算机，如单处理器的个人计算机。SIMD 计算机由在 N 个数据流上操作的 N 个处理器组成，其中每个处理器同时执行相同的指令。这类机器通常是大规模并行计算机，它们包含了许多相对简单的处理器、处理器之间的通信网络，以及一个管理处理器同步操作的控制单元。处理器可能会使用共享内存，每个处理器也可能有它自己的本地内存。串行程序需要进行很大的修改来充分利用 SIMD 架构，而且并不是所有问题都能变成 SIMD 实现。因为这些原因，在 20 世纪 90 年代流行的 SIMD 架构（如 Thinking Machine 公司的 CM-2）不再继续了。如今，这些架构仍然在数字信号处理器和视频游戏中出现。

MISD 计算机使用 N 个处理器在共享内存中操作单个数据流。每个处理器执行它自己的指令流，这样多个操作就在相同的数据项中同步执行。MISD 架构相对较少，脉动（systolic）阵列是最有名的例子。

MIMD 是最通用、最流行的一类并行架构。MIMD 计算机包含 N 个处理器、N 个指令流和 N 个数据流。处理器类似于在 SISD 计算机上使用的处理器；每个处理器有它自己的控制单元、处理单元和本地内存[⊖]。MIMD 系统通常包含共享内存，或者将处理器互连的通信网络。处理器可以在单独或不相关的任务上工作，或者它们可以合作解决单个任务，这提供了很大的灵活性。处理器交互程度很高的 MIMD 系统叫做“紧耦合”（tightly coupled）系统，而处理器交互程度较低的叫做“松耦合”（loosely coupled）系统。MIMD 系统的例子包括多进程 PC 服务器、对称多处理器（Symmetric Multiprocessor，SMP）和可扩展并行处理器。今天，在速度排名前 10 的计算机（2009 年 11 月）中[⊖]，我们找到了 IBM、Cray、SGI 和 Sun 等公司制造的计算机，也有一台是中国国防科技大学制造的。

尽管 MIMD 常常表示一个使用两个或更多相同处理器的自治的并行计算机，但 MIMD 也有*分布式计算*（distributed computing）架构的特点。在分布式计算中，由局域网
415 或广域网连接的多个计算机合作解决单个任务。尽管在分布式计算环境中处理器之间的耦合十分松散，但是 MIMD 架构的基本组件还是保留的。每个计算机包含一个处理器、控制单元和本地内存，并且局域网或广域网提供了处理器之间的通信网络。接下来将进行详细介绍。

2. 性能指标

当我们使用并行计算时，我们通常想要知道，与运行在单处理器上的串行程序相比，我们得到了什么程度的性能提高。有一些可用的指标来衡量并行算法的性能。一个这样的指标是，解决同样的问题，相对于最好的串行算法，并行算法所能得到的*加速比*（speedup），定义如下：

⊖ 在 MIMD 系统中使用的处理器可能和在 SISD 系统中使用的处理器一样，它们也可能提供额外的功能，如与共享内存相关的硬件缓存。

⊖ http://www.top500.org/。

$$S = \frac{\text{可用的最佳串行算法的运行时间}}{\text{并行算法的运行时间}} \tag{10-1}$$

在理想的情况下，当在 N 个处理器上运行一个并行算法时，我们可能会得到完美的加速比，即 $S=N$。在实践中，完美的加速比是无法取得的，因为要么问题不能被分成 N 个相同的子任务，要么并行算法会有控制开销（如调度或同步），要么问题包含一个固有的串行组件。Amdahl 法则 [4] 表明，对于一个给定的问题所能取得的最大加速比与 f 有关，f 即问题中必须串行计算的部分。这个关系由下面的式子给出：

$$S \leqslant \frac{1}{f+(1-f)/N} \leqslant \frac{1}{f}$$

另一个并行算法性能的指标是效率（efficiency），即

$$\Phi = \frac{S}{N} \tag{10-2}$$

其中 S 是加速比，N 是处理器的个数。当 $\Phi=1$，即没有任何处理器有过空闲或者执行了不必要的操作时，就达到了理想的效率。和完美的加速比一样，理想的效率在实践中也是无法取得的。

最终，并行程序相对于串行程序的性能提高将体现在完成处理任务所需的时间减少上，同时也要考虑与运行并行程序所需的并行硬件相关的额外金钱支出。这就最好地刻画了并行程序的性能和代价的总体情况。

10.4.2　在 MIMD 架构上的并行信息检索

MIMD 架构在如何定义和利用并行来解决问题方面提供了很大的灵活性。信息检索系统利用 MIMD 计算机的一个最简单的方式是使用多任务（multitasking）。并行计算机中的每个处理器都运行一个独立的搜索服务。这些搜索服务并不合作处理单个查询，但是它们可能会共享代码库和文件系统缓存的或载入共享内存的数据。将用户查询提交到搜索服务是由代理程序管理的。 416
如图 10-5 所示，它从终端用户接受搜索请求，然后将请求分发给可用的搜索服务。随着更多的处理器加入到系统中，会运行更多的搜索服务，那么就可以并行地处理更多的搜索请求，从而提高了系统的吞吐量。然而，需要注意的是，单个查询的响应时间仍然是不变的。

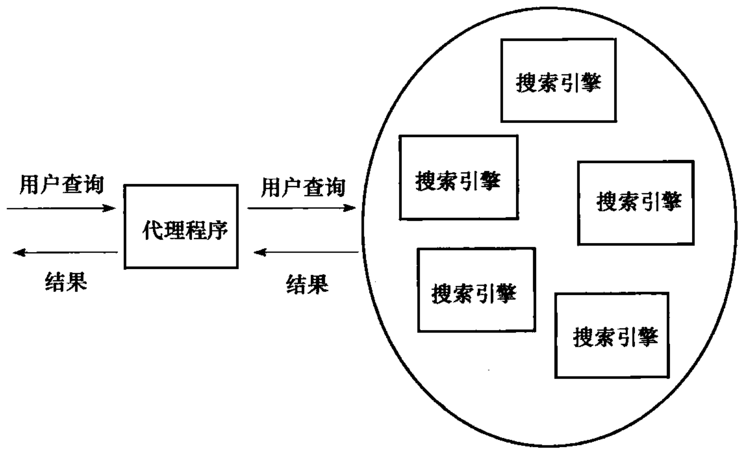


图 10-5　在 MIMD 机器上的多任务并行

尽管这个方法很简洁，但是必须注意合理地平衡系统上的硬件资源。特别地，随着处理器数目的增加，硬盘和 I/O 通道的数量也必须增加。除非内存能容纳整个检索索引，否则运行在不同处理器上的搜索进程将执行 I/O 操作，竞争硬盘访问。硬盘瓶颈对性能来说将是灾难性的，可能会抵消增加更多的处理器所期望得到的吞吐量收益。

除了给计算机增加更多的硬盘外，系统管理员必须合理地将索引数据分布到所有的硬盘上。只要两个搜索进程需要访问保存在相同硬盘上的索引数据，那么硬盘竞争的问题就还将存在。一个极端情况是，将整个索引复制到每个硬盘，这将消除硬盘竞争的问题，但代价是提高了存储空间需求和更新的复杂度。或者，系统管理员可以根据配置信息划分并复制索引数据到硬盘上；复制频繁访问的数据，而比较少访问的数据被随机地分布。另一个方法是建立一个磁盘阵列，即 RAID[365]，并让操作系统处理索引划分问题。通过将文件分散到许多磁盘上，磁盘阵列可以提供低延迟和高吞吐量的硬盘访问。

为了超过多任务的并发能力，并改善查询的响应时间，执行单个查询所需的计算必须分割成多个子任务，并且分布到多个处理器上，如图 10-6 所示。在这种情况下，代理程序和搜索进程像之前一样并行地运行在单独的处理器上，但是现在它们合作执行
417 同样的查询。在这个系统中，高层的处理是如下进行的：代理程序从终端用户接受一个查询，将它分发到所有的查询进程。然后每个查询进程执行该查询的一部分，并将中间结果传回给代理程序。最后，代理程序将中间结果合并成一个最终的结果展示给终端用户。

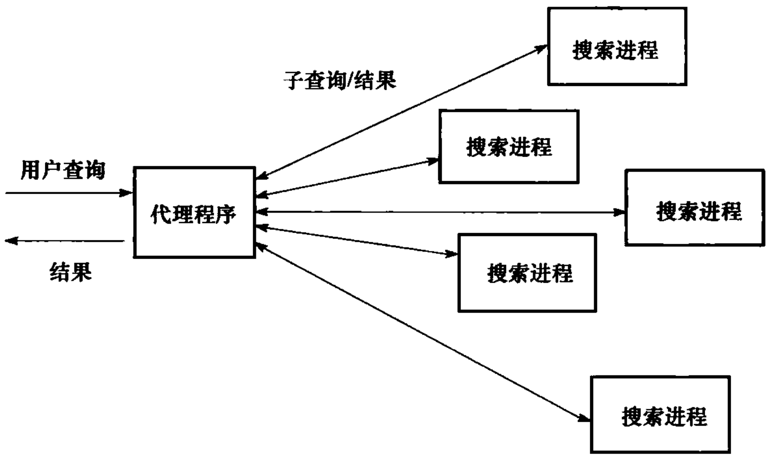


图 10-6 在 MIMD 机器上并行处理的划分

10.4.3 在 SIMD 架构上的并行信息检索

SIMD 架构只适用于比 MIMD 架构更受限的问题领域。为此，SIMD 计算机相比于 MIMD 计算机来说并不常用，我们主要是因为历史原因才包含了这一节。

或许 SIMD 架构最有名的例子就是 Thinking Machine 公司的 Connection Machine 2 (CM-2)，它在 20 世纪 90 年代就不再继续使用了⊖。CM-2 可用于支持签名文件和基于倒排索引的信息检索算法。在 CM-2 中的每个处理单元有一个位的算术逻辑单元（Arithmetic Logic Unit，ALU）和少量的本地内存。这些处理单元执行本地或者非本地的并行指令。一

⊖ 在 Thinking Machine 公司于 1994 年申请破产前最新的模型是 CM-5，它使用的是 MIMD 架构。

条本地并行指令使得每个处理单元在保存于本地内存中的数据上执行一致的操作。一条非本地并行指令涉及处理单元之间的通信，并包含对向量的所有元素求和或者找出全局最大值这样的操作。

CM-2 用一个单独的前端主机来为后端并行处理单元提供界面。前端控制后端数据的载入和卸载，并执行串行程序指令，如条件和迭代语句。并行宏指令从前端发送到后端的微控制器，它控制在后端处理单元上同时执行指令。 418

CM-2 提供了在后端处理器上的一个抽象层，叫做*虚处理器*（virtual processor）。一个或多个虚处理器映射到单个物理处理器。程序的处理需求用虚处理器表达，然后硬件将虚处理器操作映射到物理处理器上。一个物理处理器必须串行地执行每个虚处理器的所有操作。虚处理器和物理处理器的比值叫做*虚处理比*（virtual processing ratio，VP）。随着 VP 的增加，在运行时间上会出现近似线性的增加。

倒排索引

倒排索引在 SIMD 机器上较难实现。然而，Stanfill 等人［1531，1528］已经提出了两种适用于 CM-2 的倒排索引。回想 9.2 节中所介绍的倒排列表结构。在最简单的形式中，一个倒排索引为每个索引项及其出现的文档建立一个*记录*。记录是形式为（k_i，d_j）的元组，其中 k_i 是某个索引项的标识符，d_j 是文档的标识符。依赖于检索模型，记录可能额外包含权重或位置信息。如果保存了位置信息，那么就可以为 k_i 在 d_j 中的每次出现生成一个记录。

第一个面向 CM-2 的并行倒排索引实现了使用两个数据结构来保存倒排索引：记录表和索引。记录表包含来自记录的文档标识符，而索引将项映射到它们在记录表中对应的条目。在装入这些结构前记录按照索引项标识符排序。然后，按照这个排好的顺序将文档标识符装载进记录表，填入一系列长度为 P 的行中，这里的 P 是正在使用的处理器的个数。这个记录表被当做一个并行数组，其中数组下标选择某个特定的行，然后每行被分配到 P 个处理器上。对于每个项，索引保存与这个项相关的文档标识符在记录表中的第一个和最后一个条目。图 10-7 展示了一个小的文档集、原始记录，以及生成的记录表和索引。例如，为了找到包含项“piggy”的文档，我们在索引中查找“piggy”，然后知道从行 1 位置 3 到行 2 位置 1 的记录表条目包含了对应的文档标识符，即 0、1、2。

在搜索时，这些数据结构按如下过程用于对文档排序。首先，检索系统将记录表装载到后端处理器。然后，系统对查询项进行迭代。对于每个查询项，每次查找索引，返回必须处理的记录表条目的范围。然后搜索系统对包含在此范围内的行进行迭代。对每一行，包含当前项的记录表条目的处理器被激活，与之相关的文档标识符被用来更新对应文档的分数。

文档的分数是在累加器（Stanfill 称之为*信箱*）中建立的，它和记录表类似，分配在一个并行数组中。为了对特定的文档更新累加器，我们必须决定累加器所在的行和行内的位置。为了方便起见，我们假设这个信息（而不是文档标识符）保存在记录表中。而且，我们假设权重已经和每个记录联系起来，并保存在记录表中。对带权重的项打分的完整算法见图 10-8。 419

score_term 算法假设在索引中查找查询项的过程已经完成，并且结果保存在 **term** 变量中。这个算法对与查询项相关的每行记录进行迭代，并决定在当前行中处理哪些位置。**Position**是个并行整数常量，其中第一个实例是 0，第二个实例是 1……最后一个实例是 **N_PROCS-1**。根据当前行中感兴趣的位置，在 **where** 子句中激活合适的处理器。在算法的结尾

文档

This little piggy went to market.	This little piggy stayed home.	This little piggy had roast beef.

记录列表

beef	2
had	2
home	1
little	0
little	1
little	2
market	0
piggy	0
piggy	1
piggy	2
roast	2
stayed	1
this	0
this	1
this	2
to	0
went	0

索引

项	第一个条目		最后一个条目	
	行	位置	行	位置
beef	0	0	0	0
had	0	1	0	1
home	0	2	0	2
little	0	3	1	1
market	1	2	1	2
piggy	1	3	2	1
roast	2	2	2	2
stayed	2	3	2	3
this	3	0	3	2
to	3	3	3	3
went	4	0	4	0

记录表

2	2	1	0
1	2	0	0
1	2	2	1
0	1	2	0
0			

图 10-7 并行倒排索引

```
score_term(P_float Doc_score[], P_posting Posting[],
                                 term_t term)
{
     int     i, first_pos, last_pos;
     P_int   Doc_row, Doc_pos;
     P_float Weight;

     for (i = term.first_row; i <= term.last_row; i++) {
       first_pos = (i == term.first_row ? term.first_pos : 0);
       last_pos = (i == term.last_row ?
                    term.last_pos : N_PROCS - 1);
       where (Position >= first_pos && Position <= last_pos) {
         Doc_row = Posting[i].row;
          Doc_pos = Posting[i].pos;
          Weight = term.weight * Posting[i].weight;
          [Doc_pos]Doc_score[Doc_row] += Weight;
       }
     }
}
```

420

图 10-8 子程序 score_term

部分，对 **Doc_score** 的左索引提供了能够访问并行变量的特定实例。这个操作十分重要，因为它涉及处理器之间的通信。记录的权重必须从包含记录的处理器转移到包含对应文档的累加器的处理器上。当系统已经用 **score_term** 算法处理了所有查询项后，它根据文档的分数对它们排序，并返回前 k 篇文档。

把记录的权重发送给不同处理器上的累加器的代价是比较高的。为了解决这个问题，Stanfill [1528] 提出了**分区记录文件**（partitioned postings file），通过将给定文档的记录和累加器保存在相同的处理器上，消除了之前算法所需的通信代价。有两个技巧来完成这个任务。首先，当把记录装载到记录表中时，不是在行中从左向右工作，并在开始下一行前填满前一行，而是把记录添加

到对关联文档进行打分的处理器所对应的列。这保证了所有与一篇文档关联的记录将和文档累加器一起装入到相同的处理器。图 10-9a 展示了图 10-7 的记录如何能被装进一个包括两个处理器的记录表中，其中文档 0 和文档 1 分配给处理器 0，文档 2 分配给处理器 1。

图 10-9a 也展示了这个方案的一个问题。项“this”的记录是偏斜的，未贯穿连续的行。为了处理这个情况，我们应用分区记录文件的第二个技巧，就是对记录分段，使得在分区 i 中的每个索引项在字典顺序上小于或等于分区 $i+1$ 上的每个项。如图 10-9b 中使用的由三行组成的分区。注意可能需要对一些分区用空白进行填充，使之满足分区的约束。

记录表和索引在变成图 10-10 所示的最终形式前还需要经过一些更改。首先，记录列中的索引项标识符由项标签替代。系统将标签分配给项，使得在同一个分区中没有任何两个项有相同的标签。其次，在记录中的文档标识符由文档行号替代，其中行号表示哪一行包含了文档的累加器。因为累加器和记录在同样的位置（如处理器），行号已经足够来确定文档。最后，修改索引，以记录开始分区、结束分区和每个项的标签。修改的项打分算法如图 10-11所示。

a）

home	1	beef	2
little	0	had	2
little	1	little	2
market	0	piggy	2
piggy	0	roast	2
piggy	1	this	2
stayed	1		
this	0		
this	1		
to	0		
went	0		

b）

home	1	beef	2
little	0	had	2
little	1	little	2
market	0	piggy	2
piggy	0	roast	2
piggy	1		2
stayed	1	this	2
this	0		
this	1		
to	0		
went	0		

图 10-9　偏斜、分区的记录

索引

项	第一个分区	最后一个分区	标签
beef	0	0	0
had	0	0	1
home	0	0	2
little	0	0	3
market	1	1	0
piggy	1	1	1
roast	1	1	2
stayed	2	2	0
this	2	2	1
to	3	3	0
went	3	3	1

记录表

2	1	0	0
3	0	1	0
3	1	3	0
0	0	1	0
1	0	2	0
1	1		
0	1	1	0
1	0		
1	1		
0	0		
1	0		

图 10-10　分区记录文件 421

```
ppf_score_term (P_float Doc_score[], P_posting Posting[],
                term_t term)
{
  int     i;
  P_int   Doc_row;
  P_float Weight;

  for (i = term.first_part * N_ROWS;
       i < (term.last_part + 1) * N_ROWS; i++) {
    where (Posting[i].tag == term.tag) {
      Doc_row = Posting[i].row;
      Weight = term.weight * Posting[i].weight;
      Doc_score[Doc_row] += Weight;
    }
  }
}
```

图 10-11　子程序 ppf_score_term 422

这里 **N_ROWS** 是每个分区的行数。这个算法遍历分区记录文件中项所在的全部分区的所有行，然后激活匹配的记录所在的处理器。每个活跃的处理器从记录中抽取文档行，计算项权重，并更新文档的分数。在所有的查询项都处理过后，系统对文档排序，并返回前 k 篇文档。Stanfill[1528] 显示，分区记录列表文件会产生大约原文本 1/3 的空间开销（其中 10%～20%是浪费在分区填充上），同时在 TB 级别的文本上使用一个 64K 处理器的 CM-2，即能够支持快速的查询响应。

10.5 基于集群的信息检索

集群计算介于并行计算和分布式计算之间。服务器集群是一个由很多在物理上接近的计算机组成的分布式系统，并且通常通过一个快速的局域网连接。每个集群是在图 10-1 中所展示的可扩展的通用架构的基本元素。随着局域网变得越来越快，集群表现得越来越像并行计算机，虽然服务器之间的连接要比并行机器中的多处理器总线速度要慢。

与单台计算机相比，集群通常用来提高性能或可用性，它通常比单个有同等速度或可用性的计算机有更好的成本效益。集群计算的一个重要的问题是平衡服务器间的负载。为此，可以部署特殊的节点，叫做*负载均衡器*（load balancer），它负责平衡不同机器之间的负载，提高集群系统的性能。如果在同样的硬件上运行，最忙的服务器的容量将限制整个系统的容量。如果服务器运行在不同的硬件上，那么负载均衡需要使每个服务器上的负载与其硬件的速度成比例，这依赖于系统的拓扑结构，这也许会是一个更难的问题。

如今存在着许多不同类型的集群。例如，以高可用性为目标的集群有冗余节点来消除因为单个计算机故障所造成的问题。通过拥有一些能实时替换故障节点的备用节点，能够以较低的代价提高可用性。其他集群主要用于计算目的，如下面的这两种情况：

- *贝奥武夫集群*（Beowulf Clusters）是一个由相同节点组成、运行在专用网络上的集群。当计算模型需要频繁通信时，需要这类集群。
- 在*网格计算*（grid computing）中，网格负责将任务分配给计算机，每个计算机独立于集群中的其他机器执行任务。虽然硬盘等一些资源可能被所有节点共用，但一个任务的中间结果不应该影响在网格中其他节点上运行的其他任务。网格计算最适用的场景是，工作由许多独立的任务组成，它们不需要在计算过程中共享数据。也就是说，与 Beowulf 集群相反，网格计算最常用在很少或者没有内部节点通信的情况中。

我们在并行信息检索中提到的指标同样可以应用到基于集群的计算。与效率等同的叫做
423 *负载均衡*（load balancing）。也就是说，我们希望所有的机器都做几乎相同量的工作。有很多方法来衡量负载均衡。例如，我们可以计算与平均负载 l 相比最大的偏差比例：

$$LB = \max_{i=1}^{n}\left(\frac{|load_i - l|}{l}\right) \tag{10-3}$$

其中 $l = \sum_{j=1}^{n} load_j / n$。注意 LB 的范围可以是从 0（完美的平衡）到 $n-1$（完全不平衡，所有的负载都由一台机器处理了）。我们可以将其用 $1-LB/(n-1)$ 将其逆转并归一化。

负载均衡可以通过组合多种技术来实现。最简单的方法是使用一个特殊代理——负载均衡器，监管这个任务。然而，在有些情况中却无法实现，需要一些特定的负载均衡算法。

为了对集群编程，有一些中间件软件，如消息传递接口（Message Passing Interface，MPI）或并行虚拟机（Parallel Virtual Machine，PVM），它使得程序能够移植到很多集群上。另一个可能的方案是由 Dean 等人 [485] 引入的 map-reduce 并行计算范式，它在 Hadoop软件包 [691] 中作为开源软件发布。

集群计算可以用于很多用途。我们这里主要的兴趣是利用它来运行信息检索系统，特别是，利用它来运行搜索引擎。更加重要的是，因为集群计算是如今实现搜索引擎的架构。为此，我们将在 Web 检索的环境下更加详细地介绍基于集群的信息检索，见 11.4.2 节。

10.6 分布式信息检索

10.6.1 介绍

分布式系统通常由一系列服务器进程组成，每个进程在一个单独的处理节点上运行，一

个指定的代理进程负责接受客户端请求，将请求分发到服务器，从服务器收集中间结果，并将中间结果合并成最终的结果给客户端。这个计算模型十分类似于图 10-6 所展示的 MIMD 并行处理模型。这里主要的区别是子任务运行在不同的计算机上，并且子任务之间的通信是通过某个网络协议实现，如 TCP/IP[408]（而不是基于共享内存的进程间通信机制）。另一个重要的区别是，在分布式系统中，经常选择分布式服务器的一个子集来处理某个特定请求，而不是将每个请求都广播到系统中的每个服务器上。

1. **分布式计算**

分布式计算是通过联网的多台独立的计算机处理单个问题的应用，这些计算机使用消息机制来互相通信。分布式计算系统可以看成是一个 MIMD 并行处理器，它有着相对较慢的 424 进程间通信通道，并且能在系统中自由地部署不同类别的处理器。实际上，分布式系统中的单个处理节点可以是一个并行计算机，这由它自己决定。而且，如果它们都支持相同的公共界面和调用服务的协议，那么系统中的计算机可以属于不同的参与者，并由不同的参与者操作。

MIMD 并行计算机和分布式计算环境之间主要的区别是处理器间通信的代价，这在分布式计算环境中要高得多。为此，分布式程序通常是粗粒度的，而单个并行计算机上的程序往往是细粒度的。颗粒度指的是，相对于通信量，程序所执行的计算量。粗粒度程序，相对于通信量，执行较大规模的计算；而细粒度的程序，相对于计算量，执行较大规模的通信。当然，在解决问题的不同时段，一个应用可能会使用不同级别的颗粒度。

另一个重要的区别是，在基本的并行计算中，内存是共享的，而在分布式计算中，每个处理器有它自己的本地内存。另一方面，分布式系统在实际中也是个并行系统。特别地，因为在服务器中只有少量通信，我们可以将 10.5 节提到的网格计算的想法扩展成在不同地理区域、通过互联网相连的不同类型的计算机。

在分布式系统中，对可扩展性十分重要的元素有四个：划分、通信、可靠性和外部因素。*划分*处理数据的可扩展性，在大型信息检索系统中，它表示划分文档集和索引，见 10.3 节。*通信*应对处理的可扩展性，在我们的情况中是查询处理。如果一个系统的操作不会失败，那么它是*可靠的*。因而，可靠性用于评价系统的性能以及它所提供服务的可信任性。它包括可用性（availability，正确服务准备就绪）、可依赖性（reliability，正确服务的延续性）、安全性（safety，不会对用户和环境造成灾难结果）和保险性（security，同时保证只对授权的用户可用、机密性和完整性）。*外部因素*是系统的外部限制。在分布式检索系统中，有很多外部因素，从网络限制到系统开发人员的质量 [1153]。

在这样的情况下，10.5 节定义的负载均衡概念同样是重要的。

2. **目标和关键问题**

有些应用可以很好地转变成分布式实现，它们通常涉及可以分割成粗粒度操作的计算和数据，而这些操作之间有相对较少的通信。基于文档划分的并行信息检索很符合这个特点。在 10.4.2 节，我们了解了怎样用文档划分来将搜索任务分成多个自治的子任务，每个子任务进行大量的计算和数据处理，但其间只有少量通信。而且，或者出于管理的目的，或者是为了将相关的文档合并到单一来源，文档几乎总是会被组合成文档集。因此，文档集为将数 425 据分布到多个服务器，为划分计算提供了天然的颗粒度。

信息检索系统最终的目标是能够在大的文档集中快速、准确地回答查询。根据 [99]，这包含了三个不同的目标，我们接下来对此加以详细阐述。第一，信息检索系统需要应对内容增长和变化、用户数的增加以及搜索模式（用户模型）的多样性。为此，系统必须是可扩

展的。可扩展性是指，随着加入更多的资源，系统处理越来越多的工作量的能力。第二，系统必须提供较高的容量，这里的容量是指，在任何给定的时间、给定的响应时间和吞吐量目标，系统所能承受的最大用户数。第三，系统不能在答案质量方面做出妥协，因为快速输出坏的答案是比较容易的。这些主要的目标——可扩展性、容量和质量，是信息检索系统的所有模块都需要的：

- 数据收集模块依赖于系统。在某些情况下，数据已经给定，并且已经是分布式的（如搜索多个文档集）。在另一些情况下，数据虽然已经给定，但系统必须将数据分发。最后，在 Web 搜索引擎中，数据已经是分布式的，但是我们必须重新收集它们，这个动作叫爬取（见第 12 章）。
- 索引模块有两个任务：划分文档集和索引。划分过程需要找到好的分配方案来把文档集按照文档或者索引项划分给服务器。和传统信息检索系统一样，索引过程主要是建立索引结构。可以利用并行硬件平台设计和实现高效的文档索引算法，如 10.3 节所讨论的。
- 查询处理模块以可扩展的方式处理查询，并且具有这些属性：低响应时间、高吞吐量、高可用性和结果的高质量。

如表 10-1 所示，除了数据划分之外，所有模块有三个共同的高层次问题，它们对于分布式系统的可扩展性都是相当关键的：可靠性、通信和外部因素。我们先解释前两个方面，因为外部因素是比较明显的。接下来，我们讨论它们如何影响索引和查询处理，以扩展我们在本章已经介绍的内容。

426 **表 10-1 分布式信息检索系统的主要模块，以及每个模块的主要问题（基于 [99]）**

模块	通信	可靠性（同步化）	外部因素
索引	重索引	部分索引 更新 合并	内容增长 内容变化 全局统计
查询	复制 缓存	排序聚合 个性化	用户需求变更 用户基数增长 域名解析系统（DNS）

3. 可靠性

故障可能导致系统或者系统的某些部分不可用。对于关键任务系统来说，这是特别不希望看到的，如商业搜索系统的查询处理组件。特别地，可用性对这类系统常常是影响最大的属性，因为它影响了提供搜索服务的公司的主要收入来源。然而，在有很多组件的分布式系统中，我们可以利用数据的重复性用不同的方式处理故障。分布式系统中的故障虽然较少出现，但另一方面，仍然比我们所预想的更加频繁，例如 Junqueira 和 Marzullo[857] 所展示的关于站点可用性的例子。联合信息检索系统由若干分布在广域网上的查询处理器（相当于站点）组成，而在任何时候都可以断开连接的对等系统，甚至会有更多的查询处理器，因此都出现站点不可用的情况。

应对故障的一个经典的方法就是复制。在一个分布式信息检索系统中，复制体现在不同的方面：网络通信、功能和数据。为了复制网络通信，我们复制链接数，使站点多处安置。在客户端/服务器系统中，在网络通信上的这种冗余减少了部分客户端和服务器无法通信的概率。在由大量处于不同地理位置的客户端组成的对等系统中，这个问题并不那么严重，因为设计这样的系统已经在网络连接性方面提供了足够的多样性。

对于功能和数据，有两种可能级别的复制：单个站点和跨站点。在单个站点中，如果功能或数据未被复制，那么单点故障可能会造成服务无法访问。例如，如果集群有单个负载均

衡器，并且该负载均衡器发生故障，那么这个集群将因为这个故障而停止工作。使用多个站点提高了总有某个服务器能够执行请求（如处理查询）的可能性。当站点故障比较频繁时，这点尤其必要。这个悬而未决的问题是如何选择站点的位置，以及满足某个特定可用性目标所需要的复制标准。

4. 通信

正如我们之前已经介绍的，将信息检索系统的任务分布化能实现多个我们所期望的特性。将任务分布到多个服务器上的一个主要缺点是这些服务器必须进行通信。网络通信可能会成为瓶颈，因为带宽常常是稀缺的资源，特别是在广域系统中。而且，服务器之间的物理距离也极大地增加了传递特定消息的时延。尽管在局域网消息时延是百微秒的数量级，但是在广域网，它可能多达几百毫秒。

实现分布式信息检索系统的方案必须考虑到这些限制。作为一个简单的例子，假设我们将前端服务器建模成一个排队系统 $G/G/c$，模型中的 c 个服务器对应于在 Web 服务器上服务请求的线程[⊖]。如果每个线程对请求的响应依赖于这个线程和系统其他部分的通信，那么 427
带宽和消息时延将对线程回答这样一个请求所需的时间造成影响。假设 $c=150$（在 Apache 服务器上最大客户端数的典型值），图 10-12 展示了在不同的平均服务率情况下，系统容量的上限（对于给定的点（x，y），如果 x 是平均的服务时间，那么容量就必须小于 y，否则服务队列将增长到无限大）。在图 10-12 中，随着每个线程平均服务时间的增加，最大容量下降得十分厉害：当平均服务时间从 10ms 上升到 100ms 时，它从 15 下降到 2。这个简单的实验说明了在设计方案时考虑网络通信影响的重要性。

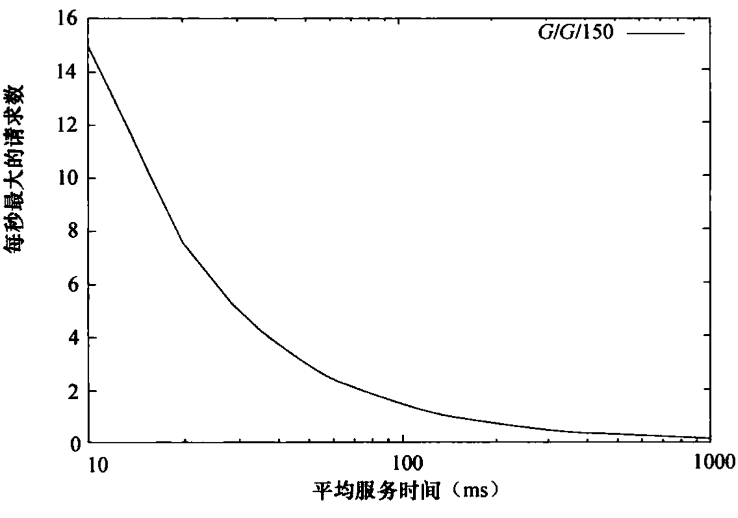


图 10-12 使用 $G/G/150$ 模型的前端服务器的最大容量，来自 Challenges on distributed Web retrieval，*Proceedings of ICDE 2007*（Baeza-Yates et al.），pp. 6-20（2007），©2007 IEEE [99]

10.6.2 索引

在查询处理器之间划分索引的一个方法是考虑文档中提到的主题 [1361]。例如，一个

⊖ $G/G/c$ 队列建模了一个系统，该系统中查询到达和服务时间的分布是任意的，并且有 c 个服务器来服务请求 [1385]。

拥有特定主题的文档索引的查询处理器也许能更加有效地处理与该主题相关的查询。根据主题分发查询涉及识别文档和查询的主题。将查询匹配主题就是在 10.3.2 节所说的文档集选择问题。文档集的所有分区根据它们和查询的相似程度排序。然后，由排序最高的那部分分区真正地处理查询。这样划分索引的一个挑战是文档或查询的主题分布的变化可能会对分布
428 式检索系统的性能造成负面影响。如 [308] 所示，对分布式查询处理架构的模拟说明，查询主题分布的变化可能会对性能造成负面影响，导致未能充分利用资源或者对热门的主题只分配了较少的资源。针对这个挑战的一个可能的解决方案是，考察来自信息检索系统的查询日志的信息，自动地重新配置索引划分。

根据查询的语言来划分索引也是一个合适的方法。识别文档的语言可以通过如下的方法进行，如比较文档和每种目标语言的 n 元文法（n-gram）语言模型 [346]，或者比较某种语言最频繁的词语出现在文档的概率 [673]。类似的技术也可以识别查询的语言，即使每个查询的文本量和额外的上下文元数据有限，但该过程也可能会引入错误。在使用语言分发查询时，另一个挑战是存在多语言文档，特别是在 Web 上。例如，技术内容的文档可能有多个英语术语，尽管主要的语言是另外一个。此外，查询也可以是多语言的，涉及不同语言的查询项。

以分布式的方式建立索引是一个具有挑战性的问题。目前为止，很少有论文提出以分布式的方式建立倒排索引的方法。例如，一种可能的方法是以管道形式组织服务器 [1117]。[485] 给出了一个在大规模计算机集群上建立索引的 map-reduce 方案。

1. 可靠性

分布式索引本身并不是一个关键的过程。然而，分布式检索系统的良好运作依赖于存在促进查询处理的索引结构。例如，如果足够多的索引服务器失效，使得无法访问索引数据来处理查询，那么服务作为一个整体也就失效了，尽管系统的其他部分可能还在很好地工作。另一个与可靠性相关的问题是索引的更新。在那些必须返回最新查询结果，以及内容更新十分频繁的系统中，很重要的一点是，保证在给定的时间点，可用的索引数据能够即时反映内容的变化。

还有一些和划分方案相关的可靠性问题。在项划分系统中，如果系统的一个服务器失效了，那么就不可能恢复那台服务器上的数据，除非它进行过复制。如果未曾复制，那么一个不高效但可行的恢复方案是重建整个索引。另一个可能方案是，使不同分区上的数据部分重叠，这样如果一台服务器失效了，那么至少其他服务器仍然能够回答查询。文档划分系统对于服务器失效问题则更加健壮。假如一个服务器失效了，那么系统仍然有能力回答查询，虽然没有使用所有子文档集，但可能并不会损失太多的效果。在分布式信息检索系统领域中，可靠性问题并未研究得很成熟。通过追踪实际的系统来进行精确的分析能更好地说明这些
429 观点。

2. 通信

如果我们发现查询的分布有变化，那么说明用户模型可能也已经改变了。处理这样的问题是重要的，因为信息检索系统可能会在偏离真实的情况下进行操作。在这种情况下，系统应该适应新的条件，也就是说索引必须再次进行划分。一个简单、直接的方法是暂停部分索引，替换它，然后重新初始化。尽管这样暂时减少了容量，然而为了使信息检索系统正确运行，这个限制并不是一个问题。

索引过程受制于分布式合并操作，并且这也会影响服务器之间的通信。取得这个目标的一个实用方法是 map-reduce 方法 [485]。然而，如果不同站点独立执行操作，那么这就需

要大量的带宽。在这种情况下，对于这种合并操作的好机制必须同时考虑通信和计算两个方面。

实用的大型信息检索系统通常在每次文档集更新后，从头开始重建索引。但对于新闻文章和博客等一些更新十分频繁的特殊文档集来说，并不能这样更新，而通常需要某种在线索引维护策略。因为更新操作常常需要锁住索引，所以动态的索引结构限制了系统的容量和响应时间，这样可能危害整个系统的性能。一个十分有趣的问题是，是否有可能安全地锁住索引，同时又不损失太多性能。这在项划分的分布式信息检索系统中更易成为问题。那些需要频繁更新的索引项可能分布在多个不同的服务器上，这样就放大了加锁的影响。

3. 外部因素

在分布式信息检索系统中，有多个瓶颈需要处理。取决于如何划分索引，不同的因素可能会导致一些严重的问题。例如，在文档划分的信息检索系统中，有可能需要计算一些全局参数的值，如索引项的文档集频率或反比文档频率。有两种可能的解决方法。一种方法是，通过在索引过程后聚合所有的本地统计数据，可以计算最终的全局参数值。而采用另一种方法，最终聚合操作经常也有可能避免。在这种情况下，计算全局统计数据的问题就转移到了系统代理上，它负责将查询派发给查询处理服务器并合并结果。

为了计算这些统计数据，代理程序通常用一个两轮协议来处理查询。在第一轮中，代理程序从每个服务器上请求本地统计数据；在第二轮中，它将全局统计信息附加到包含查询的第二个消息上，再从每个服务器请求结果。这时的问题是，这样一个使用本地而不是全局统计的“智能”的划分策略，对最终的系统有效性会有什么样的影响？回答这个问题十分困难。在真实世界的搜索引擎中，实际上很难定义一个查询的正确答案是什么，故而难以知道使用本地统计数据是否会造成影响。衡量这个效果的一个可能的方法是比较在全局统计数据 430
上计算的结果集和只在本地统计数据上计算的结果集。而且，如果我们利用了文档集选择策略，那么全局统计数据就无法使用了。

10.6.3 查询处理

以分布式的方式处理查询意味着，当处理某个特定查询时，需要决定从分布式系统分配哪些资源。在分布式搜索系统中，可用的资源池由承担如下角色之一的组件组成：协调器、缓存或查询处理器。协调器从客户端计算机接受查询，并决定如何将这些查询分发到系统的不同部分，以便查询可以在那里被恰当地处理[⊖]。查询处理器持有索引或文档信息，它们分别用于检索和准备结果展示。

网络通信是不同参与者需要通信来进行工作的这类分布式系统不可或缺的一部分。根据网络的类型和物理接近性，时延的变化范围可能是相当可观的，所以在处理单个查询时涉及多个服务器代价可能会比较高。为了缓解这个问题，缓存服务器可以为最常访问或最热门的查询保存结果，协调器可以使用缓存的结果回答客户端。在通信代价比较高的情况中，通过只接触单个缓存服务器从而回答某些查询的这个简单方法，缓存服务器可以减少查询时延和服务器负载。

一个重要的假设是，有一个或多个服务器实现每种角色的组件。这个假设对于大型系统——高访问量和大数据量的系统来说尤其重要。在设计组件时，我们可以加入更多的物理

⊖ 协调器可以是文档或项划分的分布式系统的代理程序，或者是将查询分发给不同站点的站点级别的代理。因此，我们使用一个更加通用的术语，而不叫它代理程序。

服务器来提高整体的系统容量，这对于这样的大型系统是十分重要的，因为这使得系统具有可扩展性。实际上，将系统的多个部分分割成多个组件角色已经是提升扩展性的一种尝试，因为单个独立系统不可能不受限制地扩展（想象 Web 搜索引擎的大小）。因为这些服务器可能存放在不同的物理位置和地理区域，所以我们将每组服务器叫做一个站点（site）。

图 10-13 描绘了一个多站点分布式信息检索系统的实例，它由上面所述的组件和对应角色组成。在不同的区域有三个站点。每个站点由多个协调器、缓存和查询处理器组成。来自客户端的查询被导向到最近的服务器，这里是站点 A（路径 1）。站点 A 的协调器因为负载均衡或文档集选择等原因将查询发送给站点 B（路径 2）。站点 B 在它的处理器上回答查询问题，并将结果返回给客户端（路径 3）。

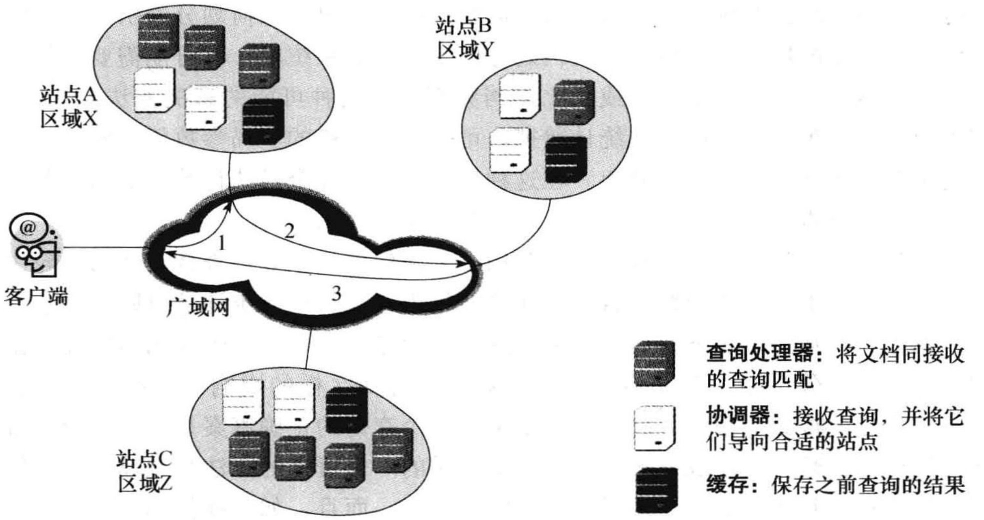


图 10-13 一个分布式查询处理系统的实例。每个服务器的角色在右侧描述，来自 Challenges on distributed web retrieval，*Proceedings of ICDE 2007*（Baeza-Yates et al.），pp. 6-20（2007），©2007 IEEE [99]

我们根据四个属性对分布式查询处理系统分类：

- **组件的数目**。对每个角色，系统可以有多个组件。有多个协调器可以改善响应时间
431 和可用性，提升用户的体验。对多缓存组件有同样的结论：它们同样有改善响应时间和可用性、减少查询处理器的服务器负载的潜力。缓存的可用性不仅需要处理缓存组件的失效，而且还包括查询处理器的失效。如果某个查询处理器暂时不可用，那么缓存服务器就可以在断供期提供缓存的查询结果。最后，由于地理位置和资源的多样性，多个查询处理器使得系统更加可靠，并且提供了可扩展的解决方案。
- **可连接性**。所有组件要么连接到相同的局域网，要么分布在不同的地理位置，通过广域网连接。
- **角色的差别**。一个查询处理系统的组件可以有一个或多个角色。通常，组件（协调器、缓存和查询处理器）实现服务器端，用于处理查询，而客户端只负责提交查询。这实现了传统的客户端/服务器模型，因为在客户端和服务器之间有着很明显的差别，前者只提交查询，后者只处理查询。另外，它们也可以同时是客户端和服务器，如在对等（P2P）系统中［181，443，1559］。在这样的系统中，所有的端点都承担我们上面提到的所有角色。

- **交互**。在联合（federate）系统中，独立的实体形成单个系统（见 10.7 节）。例如，一个跨越多个国家的组织可能有多个独立的系统，它们一起形成了整个组织的系统。联合系统也可能由不同组织的站点组成，并且用一些形式协议来限制每个站点只能执行某些特定的行为。在联合系统中的交互更加简单，因为有理由假设某些实体会监管这个系统。于是组件就可以互相信任，可以访问任何其他组件，获取所需的信 432 息，并且假设所有其他组件都以最有利于系统的方式来运作。然而，在*开放系统*中[1]，情况可能并不是这样的 [1473]。来自不同组织的站点虽然有合作，但并不统一，因此可能会从利己的角度操作，如改变查询解析的优先级，从而影响特定查询的执行性能。

组件的数目是重要的，因为它决定了可用于查询处理的资源量。根据这些组件如何连接（局域网与广域网），如何分配这些组件的选择也会变化，因为不同的选择会导致不同的性能。实际上，最小化每个查询所用的资源量一般来说是比较重要的目标，因为如果每个查询使用更少的资源，那么系统的整体容量就会提升。在客户端/服务器系统中，服务器端可用的资源量决定了系统的总容量。这样，处理查询的所有可用资源量并不随着客户端的数目而增长。然而，在对等系统中，任何新的参与者同时是新的客户端和新的服务器。假设“免费搭便车”并不流行，那么处理查询的所有可用总资源量随着客户端的数目增长而增长。在联合系统中，独立的系统组合形成单个系统，因此不需要考虑伙伴间的信任和正确行为等问题。在开放系统中，伙伴关系有可能提高系统提供给客户端的总体服务质量。然而，在这样的系统中，伙伴可能会以利己的方式来分配资源，因此可能会对某一方获得的结果造成负面的影响。

因为所有的系统都划分数据，所以这实际上潜在地增强了查询的吞吐性能。在按文档划分的情况中，我们可能只选择搜索服务器中实际包含相关文档的那部分机器，而不是使用系统中所有可用的资源来执行查询。这个选择的子集可能只包含一部分相关文档集。然而，尽可能获得最大部分的相关文档是一个具有挑战性的问题，通常叫做*文档集选择*（collection selection）或*查询路由*（query routing）（见 10.3.2 节）。这也依赖于划分文档集时所使用的技术（见 10.3.1 节）。在按索引项划分的情况中，有效的文档集选择并不是一个困难的问题，因为其方法是直截了当的，即选择包含特定查询项信息的服务器。当接收到一个查询时，我们将只把查询转发到那些负责维护相关查询项子集的服务器。在按文档划分的情况中，问题复杂得多，因为我们无法提前知道哪些服务器包含了最相关的结果。

1. 查询负载均衡

实际上，查询吞吐量的主要问题是服务器负载的不平均分布。图 10-14（来自 [1673]）展示了文档划分系统（左）和管道式项划分系统（右）中八个服务器的平均忙时负载。两张 433 图中的虚线对应所有服务器的平均忙时负载。在项划分系统的情况中（使用管道架构），很明显在服务器负载分布的平衡上有所缺失，这对系统的吞吐量会有负面的影响。

为了解决这个问题，可以尝试使用“智能”划分技术，它将对索引访问模式加以估计，以此来平均分配服务器负载。在将文档随机划分到服务器的情况中，所有的服务器接收所有的查询。原则上这是一个完美的负载均衡。但是，每个服务器的工作量并不一定是相同的，因此随机划分并不保证均等的查询负载均衡 [83]。

---

[1] 开放系统在文献中也叫非合作系统。

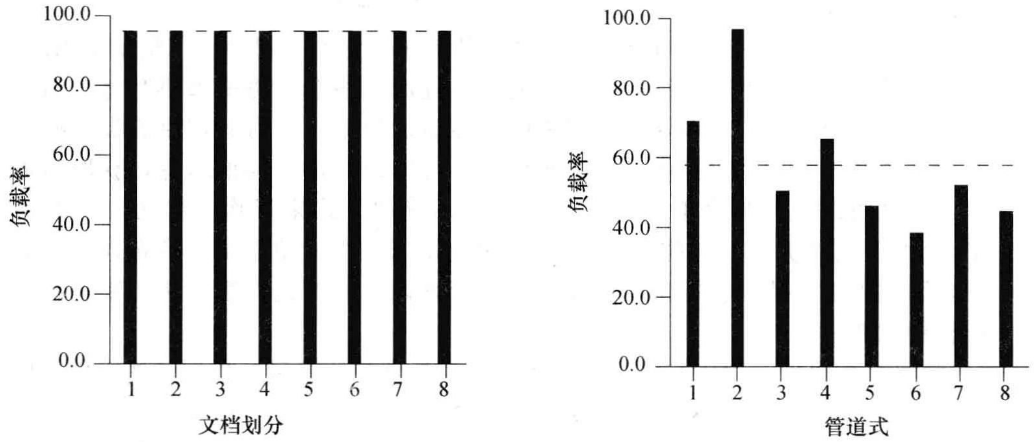


图 10-14 在文档划分系统和管道式项划分系统中，每个处理器平均负载的分布

对于项划分系统，Moffat 等人［1151］显示，通过利用索引项出现在查询中的频率信息和记录列表备份有可能平衡负载。简单地说，他们将在项划分系统中划分词汇表的问题转化为一个装箱问题（bin-packing problem），其中每个箱子代表一个分区，每个项表示为要放到箱子中的物体。每个项有一个权重，与其在查询日志中出现的频率和记录表的长度成比例。这项工作显示，该策略对项划分系统的性能是有好处的，因为它能更加平均地把负载分布到每个服务器上。不过，实验结果显示，即使考虑因负载均衡所带来的性能改进，项划分系统取得的吞吐量还是比文档划分系统低。类似地，Lucchese 等人［1060］在之前的装箱方法的基础上，设计了一个关于项和划分的权重函数，它能够对每台服务器上的查询负载建模。在原始的装箱问题中，我们简单地关注于平衡分配给箱子的权重。然而，在这个情况中，目标函数同时依赖于分配给项的权重（我们的目标），以及查询中项的共现情况。这个函数的主要目标是将查询中共现的项分配给相同的索引分区。这对减少需要查询的服务器数
434 量和每个服务器上的通信代价都是重要的。

注意，基于文档的主题或语言的划分方案可能会带来类似的查询处理器负载不均衡现象，尽管当可以预测负载时，给站点分配相应的负载是个可行的方案。

2. 可靠性

查询处理器是一个关键的角色，因为如果没有处理能力和它们所保存的数据，那么系统就无法满足客户端请求。同样，由于它们处理大量的数据（如索引和文档等），确定好的查询处理器复制方案是有挑战性的。通过在不同的查询处理器间复制数据，提高了存在某个可用的处理器包含处理特定查询所需数据的可能性。通过相互间的完全复制，可以使所有查询处理器都保存相同的数据，这样就会取得最高水平的可用性。但是，这可能会付出巨大却不必要的代价，还会减少总的存储容量。因此，一个悬而未决的问题是如何复制数据，使得系统在最小的存储代价下取得足够的可用性。

由于地理位置和资源的多样性，多个查询处理器使得系统更加可靠，并提供了一种更可扩展的解决方案。此外，缓存服务器的可用性不仅需要处理缓存组件的失效，而且也包括查询处理器的失效。如果某个查询处理器暂时不可用，那么缓存服务器就可以在断供期提供缓存的查询结果。

对于在线系统，尽管高可用性是十分重要的目标，但它并不是唯一的目标。一致性同样很重要。特别地，当我们考虑个性化等特征时，每个用户有自己的状态空间，其中包含表明

其偏好的变量；并且在每次查询时都可能会更新用户状态。在这样的情况中，有必要保证在每次更新时状态是一致的，并且用户的状态不会丢失。有一些来自分布式算法的技术，如状态机复制 [1440，971] 和主从备份 [293，1775]，可以用来实现这种容错服务。主要的挑战是将这些技术应用到大规模系统中。根据应用的需求，还有可能放松一致性的强约束，通过使用接受过时结果的技术来实现一致性的弱约束 [1407]。

通过使用缓存，还有可能进一步提高容错能力。当查询处理器失败时，系统返回缓存的结果。因此，在设计系统时可以考虑将缓存系统看做是复制的替代方案或补充方案。一个重要的问题是，如何设计这样一个能有效解决故障的缓存系统。当然，好的设计方案也应该考虑缓存系统的主要目标，即减少系统的平均响应时间、均衡执行查询的服务器负载和提高带宽利用率。这三个目标将转化为更高的点击率。有趣的是，更高的点击率也可能提高容错能力。不同于减少平均时延的目标，当处理故障时，对特定查询提供可用的结果也是重要的。例如，一个可能的缓存架构是，通过广域网使用消息进行通信的多个缓存组件。由于缓存组件间的消息时延较高，因此这样的架构并不一定能改善查询处理时延。Wolman 等人 [1715] 认为，协作的 Web 缓存机制并不一定能改善总体的请求时延，主要因为广域网通信削弱了更大的用户群所带来的好处。对于可用性，这样的架构则增加了系统所能响应用户查询的结果数，因此使得系统的可用性提高。实际上，支持使用分布式协作缓存方案来提升大型分布式信息检索系统的可用性的一个重要论据是，在广域系统中，网络的连接性常常依赖于网络提供者，并且路由错误发生得足够频繁 [1248]。 435

3. 通信

分布式查询处理架构需要考虑由通信和合并来自系统不同组件的信息所产生的代价。一个管道式项划分系统需要把经过部分解析的查询在服务器间传送 [1673，1151]。然而，当在邻近或短语搜索中使用位置信息时，服务器间的通信开销会极大地增加，因为它包括查询项的位置信息和经过部分解析的查询。在这样的情况下，位置信息需要高效地压缩，或许需要对有可能出现在查询中的词的位置进行不同方式的编码。

在文档划分的情况中，查询处理器将查询结果发送到协调器，它对结果进行合并，检出排名最高的结果，并展现给用户。当把来自大量查询处理器的结果进行合并时，协调器可能会成为瓶颈。在这样的情况中，可能使用多层的协调器来减轻这个问题 [308]。而且，文档划分系统的响应时间依赖于最慢组件的响应时间。这个限制并不一定是由它的文档分布所造成的，而且它可能依赖于磁盘缓存机制、内存量和服务器数量 [83]。

当多个查询处理器参与解析查询时，通信时延可能会比较大。减轻这个问题的一个方法是采用增量式的查询处理方法，其中更快的查询处理器提供初始的结果集。其他一些较远的查询处理器以更高的时延提供额外的结果，使得用户不断地得到新的结果。增量查询处理对结果的合并过程有影响，因为时延，更多相关的结果可能会延迟出现。在增量查询处理中，客户端使用搜索的方式也可能发生模式转变 [272]。例如，我们可以想象这样一个应用，它根据上下文自动推断出查询，并返回结果，而不需要用户直接在 Web 界面中搜索。

当查询处理涉及结果的个性化时，在搜索时就需要来自用户配置的一些额外信息，使得搜索结果适合用户的兴趣。查询处理架构并不将这些信息看成是必不可少的部分 [151]。另外一个和大型信息检索系统个性化相关的挑战是，每个用户配置表示一个状态，这必须是最新的状态，并在复制之间保持一致性。或者，一个系统可以将个性化实现为客户端上轻量级的一层。最后的这个方法很吸引人，因为它解决了集中保存用户及其行为等信息的隐私问题。但它也限制了用户只能始终使用同样的终端。 436

用户模型变得不准确是其自身的问题。因为用户行为随着时间变化，需要对模型相应地进行更新。一个简单的方法是安排模型以固定的时间间隔更新。现在的问题是，我们需要多频繁地更新它。回顾一下，更高的更新频率意味着更高的网络流量以及更低的查询处理能力。理想地，系统通信能够第一时间适应当前模型的变化。

此外，在 Web 搜索引擎等大型信息检索系统中，每天都有几十万到几百万次的查询。将记录对定义用户模型十分重要的行为，并且高效地使用它们是很有挑战性的，因为数据量可能极大。实际上，由于带宽的限制，将这些数据从一台服务器转移到另一台服务器也几乎是不可能的。

4. 外部因素

大型信息检索系统的设计在不同的方面都涉及用户（或客户端）。例如，为了评价信息检索系统的准确率，可能会建立一个相关性模型。类似地，缓存策略的设计和分析需要用户的信息，或者用户模型 [544，1002]。然而，用户行为是一个外部因素，它不受信息检索系统控制。用户搜索行为的任何重大变化都可能对系统的准确率或效率产生影响。例如，用户搜索的主题在过去已经发生了缓慢的改变 [1519]，因此可能必须要对系统资源进行重新配置来保持好的性能。在用户行为方面的变化也可能影响到缓存策略的性能。因此，如果用户行为变化得足够频繁，那么有必要提供一个能够自动重配置系统，或者简单地替换模块的机制。那么，挑战是，如何在线地确定何时用户行为发生了较大的变化。

10.6.4 Web 问题

在 Web 上的信息检索将在第 11 章展开讨论。为了完整起见，我们在这里简单地介绍如何把并行和分布式信息检索应用到 Web 上。最直接的应用是将 Web 上所有的文档收集到一个单一的大型文档集中。然后，就可以把 Web 当做一个大型的文档集，直接应用上面描述的并行和分布式的技术。这也是当前流行的大部分 Web 搜索服务所采用的方法。

另外，我们可以利用组成 Web 的分布式计算机系统，将收集、组织和搜索所有文档的工作分布开来。这是 Harvest 系统 [244] 以及新的分布式 Web 搜索架构所采用的方法。Harvest 系统由多个组件组成，它们收集、汇总、复制、分发和搜索文档。用户查询由代理程序（broker）处理，它收集和完善来自收集器（gather）和其他代理程序的信息。某个特定代理程序的信息常常与某个有限的主题集合相关，使得用户能够把他们的查询发送到最合
437 适的代理程序。一个中央代理注册点帮助用户找到对于他们的查询最合适的代理程序。

10.7 联合搜索

联合搜索系统依赖于多个不同的异质服务器来回答用户查询。因此，为了建立联合信息检索系统，我们需要考虑和许多分布式系统共有的工程问题，以及信息检索特有的算法问题。关键的工程问题基本有三个：1）定义发送请求和结果的搜索协议；2）设计一个服务器，它能高效地接收请求，初始化子进程或线程来服务该请求，并使用合适的缓存技术来利用处理过程中固有的局部性；3）设计一个中间代理，它能并行地提交异步的搜索请求给多个服务器，并将中间结果合并成一个最终的终端用户响应。算法问题也有三个：1）如何将文档分发到分布式的搜索服务器上（见 10.3.1 节）；2）如何选择应该由哪个服务器接收特定搜索请求（见 10.3.2 节）；3）如何处理请求，并把来自不同服务器的结果合并起来。我们主要关注最后一个问题。

搜索协议明确说明了在客户端和服务器间传输的消息的语法和语义，建立连接并进行搜

索所需的消息序列，以及发送消息的内在传输机制（如 TCP/IP）。这样的协议至少应该允许客户端：

- 获得关于搜索服务器的信息，如服务器可搜索的数据库列表，以及可能的统计信息。
- 使用明确定义的查询语言对一个或多个数据库提交搜索请求。
- 接收明确定义格式的搜索结果。
- 获取搜索结果所给出的条目。

对于由同样的搜索服务器组成的封闭系统来说，定制的搜索协议可能是最合适的，特别是如果需要专门功能的话（如对请求和结果加密）。另外，可以使用标准的协议，使系统和其他搜索服务器更加容易交互。用于客户端/服务器信息检索的 Z39.50 [1023] 标准（见 7.1.4 节）定义了一个广泛使用的协议，它有足够的功能性来支持大部分搜索应用。另一个用于分布式异质搜索的协议叫做斯坦福互联网元搜索协议（Stanford Proposal for Internet Meta-Searching，STARTS）[664]，它是斯坦福大学与一些搜索产品和服务提供商合作开发的。STARTS 从一开始就设计成支持联合信息检索，并包含用于解决相关算法问题的特性，如把来自异质信息源的结果合并。关于元搜索的更多信息将在 11.10.3 节和 15.3.8 节介绍。

与建立高效客户端/服务器系统有关的其他工程问题已在文献中广泛涉及（如 Comer 和 Stevens[408]、Zomaya[1800]）。接下来我们并不会回顾这些问题，而是更加详细地介绍联合信息检索的查询处理问题。 438

查询处理

在联合信息检索系统中的查询过程按如下进行：

1）选择文档集进行搜索；

2）将查询分发到所选择的文档集上；

3）在每个分布式文档集上并行地处理查询；

4）将来自分布式文档集的结果合并成一个最终结果。

正如之前所描述的，如果查询总是广播到系统中的每个文档集上，那么步骤 1）可以去除。否则，使用之前所描述的某个选择算法，将查询分发到所选择的文档集上。然后参与搜索的每个服务器在所选的文档集上使用它自己的本地搜索算法处理查询。最后，对结果进行合并。

到此为止，除了结果合并之外，我们已经涉及了所有内容。对于结果合并，有这样一些场景：如果查询是布尔类型的，并且搜索服务器返回布尔结果集，那么简单地合并所有的结果集来生成最终的结果集；如果查询涉及自由文本排序，那么就有一些从简单/朴素到复杂/精确的技术。

最简单的方法是，使用循环交错的方式来合并排好序的命中列表。这可能会产生质量差的结果，因为来自较不相关文档集的返回结果与来自高度相关的文档集的返回结果被等同看待了。对这个过程的一种改进是，基于它们的相关分数合并命中列表。但是，如在 10.3.3 节所描述的文档划分的并行过程一样，除非使用合适的全局项统计数据来计算文档分数，否则我们可能会得到不正确的结果。如果文档是随机分布的，并且全局项统计数据在所有的分布式文档集中是一致的，那么基于相关性分数的合并就已经足够用于维护检索的有效性了。然而，如果分布式文档集是按语义划分或者由不同的组织维护，那么就必须执行重排序。

Callan [323] 提出了根据加权文档分数对文档重排序，加权文档分数基于在源选择步骤时计算的文档集相似度。文档集的权重计算如下：

$$w = 1 + |C| \times \frac{s - \bar{s}}{\bar{s}} \tag{10-4}$$

其中 $|C|$ 是所搜索的文档集个数，s 是文档集的分数，而 $\bar{s}$ 是全部文档集的平均分数。

合并排序列表最精确的技术是使用精确的全局项统计数据。这可以通过很多方法完成。首先，如果出于源选择目的，已经索引了文档集，那么该索引将包含所有分布式文档集中的全局项统计数据。当代理程序将查询分发到远程的搜索服务器时，它可以在查询中包含统计数据。其次，服务器在处理过程中可以考虑这些统计数据，生成可以直接合并的相关性分数。如果没有文档集索引，那么可以用两轮通信来进行查询分发。在第一轮中，代理程序分
439 发查询，并从每个搜索服务器收集文档集统计数据。这些统计数据由代理程序合并，并在第二轮发回搜索服务器。

最后，搜索协议可以要求搜索服务器返回全局的查询项统计和每篇文档的查询项统计 [664，909]。然后，代理程序可以自由地选择查询项统计和排序算法来重排序每篇文档。最终的结果是一个命中列表，这个列表包含来自分布式文档集的文档，这些文档排列的顺序和所有文档都索引在单个文档集中的情况是相同的。

10.8 在对等网络中的检索

如果每个端点是任意的一台计算机（通常是个人的），那么当连接到互联网时，它在不可预测的时刻加入对等（P2P）网络，并维持不定长的时间。这样的网络符合对等系统。在 P2P 系统中检索数据（如文档）必须基于一个公共的平台，它充分利用分布在互联网上的资源，特别是文件共享。最早的文件共享系统，如 Napster、Gnutella 和 Freenet，其区别在于如何找到端点数据。Napster 使用中央索引服务器，是最高效的系统，但也是对攻击最为脆弱的一个。另一方面，Gnutella 使用并不高效但高度容错的泛洪（flooding）查询模型。Freenet 使用一个更加高效的启发式方法，但是并不保证能够找到文件。

这个问题的解决方案是*分布式散列表*（Distributed Hash Table，DHT）。DHT 是一个提供下列特性的中间件层：

- 分散化：端点共同形成系统，而没有任何中央调度。
- 可扩展性：即使有上百万的端点，系统也能平稳地工作，就像互联网中的情况。
- 容错性：系统尽可能可靠，即使端点不断地加入、离开和失效。

DHT 使用一个更加结构化的基于键（key）的路由方法来实现 Gnutella 和 Freenet 的分散化，以及 Napster 的效率和可保证的结果。达到这些目标的一个关键想法是，每个端点只需要和网络中其他少数端点进行协调（对于一个目前有 n 个端点的系统来说，通常是 $\Theta(\log n)$ 个）。这减少了当一个端点加入或离开网络时所需要做的工作量。此外，DHT 必须处理分布式系统的一些典型问题，如负载均衡、数据一致性和性能（特别是查询路由和数据存储或检索）。

DHT 基于一个抽象的数字键空间，其中的键有许多位，可识别网络上的任何资源（如端点、文件等）。那么，使用某种划分方案，键空间的所有权就在参与的端点划分。于是就由*覆盖网络*（overlay network）连接这些端点，使得它们能够找到键空间中任何给定键的所有者。最开始的四个 DHT 差不多是同时提出的：CAN[1336]（内容寻址网络）、Chord
440 [1537]、Pastry[1391] 和 Tapestry[1779]。它们提供不同的划分技术和路由方案。如今，典型的 P2P 共享软件，如 BitTorrent，都使用这种技术。

这些划分方案通常会采用一致性散列算法 [875] 的某些变种。一致性散列定义了一个

键间的距离函数 δ，它不依赖于地理距离和互联网的网络拓扑或网络时延。那么，用 ID（键）i 识别的端点将拥有那些根据 δ 距离函数离键 i 最近的键。一致性散列有一个重要的性质，即移除或增加一个端点只会改变那些有相邻 ID 的端点所拥有的键，而其他端点不受影响。因此，这减少了当一个端点加入或离开网络时所需做的工作。然而，仍然需要优化键空间的重组，以支持高频率的抖动（churn）⊖，因为移动保存在 DHT 中的数据将在覆盖网络中的邻居端点上产生密集的流量。

覆盖网络基于一个路由表实现，其中每个端点维护一个邻居端点的集合。这些 P2P 链接定义了某个特定的网络拓扑，形成了覆盖网络。它的主要基本性质是，每个端点或者拥有特定的键 k，或者根据之前定义的距离函数 δ，有一个更靠近 k 的所有者的邻居端点。注意到，δ 可以衡量覆盖网络中端点之间的距离。有了这个性质，如果我们并不拥有这个键，那么很容易地使用简单的贪心算法，即将消息转发给 ID 最接近 k 的邻居端点，将一个消息路由给 k 的所有者。这个算法保证了找到 k 的时间受覆盖网络的直径限制，但这并不一定是最优的。

使用 DHT 来存储和检索的典型过程如下。为了存储一个给定文件名的文档，我们在文件名上使用散列函数来生成键 k。然后我们将消息（k，文档）发送给整个系统，消息将会在覆盖网络中的邻居端点间转发，直到它到达在键空间划分中指定的负责该键 k 的端点；二元组（k，文档）最终就保存在那里。为了检索文档，我们反转这个过程：通过对文件名进行散列，端点可以找到 k 并发送一个查询来找出网络中与 k 相关的数据。这个消息将再次在覆盖网络中被传递到负责 k 的端点，反过来它直接将保存的与该键相关的数据发回来。

检索的主要缺点是，DHT 只支持完全匹配的搜索，而不是关键字搜索（无论是数据的属性还是它的内容）。因此，接下来的挑战就是将该功能加入到基础系统中。许多 P2P 网络组织已经研究了 P2P 信息检索，特别是全文检索。在非结构化网络中的搜索技术通常基于广播，因此会遭受高带宽消耗的问题。于是，已经提出了基于随机游走的方法、基于内容的路由索引和层次化网络的解决方案，以减少 P2P 网络中产生的流量。

许多 P2P 信息检索方法采用了*端点级*的文档集描述来找出可以处理查询的候选端点。这些描述有助于端点选择过程，以及接下来在所选端点上进行的文档级别的检索。通常会使用一些资源选择算法，如 CORI[320] 或基于 Kullback-Leibler 距离的算法［1731］，来选择包含与查询相关的大量文档的一小部分资源。根据它们返回相关文档的可能性对资源进行排序，选择排名最高的资源。 441

例如，［1053，1054］所描述的联合搜索系统使用了一个层次化的 P2P 网络结构。这个 P2P 网络由多个*枢纽*（hub）组成，它们根据聚类算法连接到叶子端点。查询被提交到一个或多个初始选择的枢纽端点。枢纽使用它的资源选择算法将查询发送到它的叶子端点上，并根据其他邻近枢纽的描述，发送到这些端点上。将一个生存时间（time-to-live，TTL）计数器加到每个查询上，用来限制资源使用。叶子端点在它们本地的文档集上执行查询，并将结果传回查询发送者。枢纽执行结果合并算法，聚合来自不同叶子端点的答案。［1055］改进了资源选择算法，它对历史用户行为建模，从而将搜索导向网络中合适的那部分。

Minerva[179，180］维护一个全局索引，索引中有结构化覆盖的端点选择统计数据，用于帮助端点选择过程。全局索引只保存经过压缩和聚合的、关于端点本地索引的元信息，这些元信息是用户愿意公开的。根据端点公开的关于每个查询项的元信息，初始查询者选择一些最有希望的端点。接着，它将完整的查询转发给所选择的端点，它们在本地执行该查询。

⊖ 端点加入、离开或失效的频率。

Minerva∞[1126] 是一个 P2P 信息检索系统，它基于一个保序的 DHT。它依赖于项索引网络（Term Index Network，TIN），其中索引项的全局倒排索引保存在多个端点上。查询通过一个涉及 TIN 内外端点的并行 top-k 算法来处理。

[1124] 已经发现了索引项共现统计的重要性。在这个方法中，索引项的共现信息有助于识别出最有可能的端点级别的索引条目，该条目与索引项的组合相联系。作者表明，这样的技术极大地改善了端点选择过程，以及最终的检索性能。[1792] 讨论了在出版和订阅场景中使用关键词的相关性统计。

与端点级别解决方案相反，文档级的索引方法，由于其更好的索引颗粒度，有可能产生更好的检索质量，但是需要更高的索引维护代价。这样的方法通常使用项划分方案，将整个索引分布在一个结构化的 P2P 网络上。例如，一个维护项划分全局索引的 DHT 提供了一个直接的 P2P 信息检索方案，通过将查询项散列为 P2P 的键解决查找单个查询项的问题。假设分布式索引保存了文档集中所有项的记录列表，那么就可以通过对所有查询项的记录列表交集来处理多查询项的查询。然而，这个方法面临严重的可扩展性问题，它是由对大型记录列表求交集所需的流量代价所造成的。因此，已经提出了一些方法来解决这个问题。

因为大型记录列表是这些解决方案主要关注的问题，所以 [1348] 和 [1541] 已经提出了 top-k 记录表联接、Bloom filter[213] 和缓存等作为减少多项查询的搜索代价的潜在技术。事实上，据 [362] 报告，通过将最优的 Bloom filter 设置应用到基于 DHT 的全文检索，能减少 73%的流量。然而，[1774] 的研究显示，即使能组合一些成熟的协议来减少检
442 索代价，单一项索引在实际中并不能扩展到 Web。

为了避免分布式地对大型记录列表求交集，研究人员提出了多个依靠对多项组合建立索引的方法。通过系统化地将大型记录列表截成固定的大小，HDK 方法（Highly Discriminative Keys）[1285] 的索引能减少检索的流量，同时通过对选择的项组合建立索引，可以补偿所导致的信息丢失。因此，索引包含了更多的条目，但是每个条目与更短的记录列表相联系。然而，索引条目的数量可能仍然变得很大，所以一个叫做查询驱动索引（Query-Driven Indexing）的方案就被提了出来 [1488] ——它只索引那些当前在用户查询中常见的项组合，与 HDK 方法相比，它能将索引的大小减小几个数量级，而代价只是检索质量的微小损失。类似地，通过缓存多个项查询的完整结果，分布式缓存表（Distributed Cache Table，DCT）方法 [1486] 在运行时生成索引（或分布式缓存）。

受 [542] 的算法启发的 top-k 查询处理方法已经被许多 P2P 信息检索方法所采用，用来解决大量带宽消耗的问题。主要的想法是，尽可能早地结束对查询的处理，同时保证（或提供概率保证）目前得到的排名前 k 的结果是正确的。当使用倒排索引处理多个项查询时，如果只需要交集的前 k 个，那么就没有必要扫描整个记录列表。相反，记录列表可以根据分数排序，并且有可能通过只查看在记录列表靠前部分的文档就可以得到排名前 k 的查询结果。提早终结查询对于分布式记录列表求交集是特别有用的，因为它有减少带宽消耗的直接效果。用于 P2P 网络的 top-k 查询处理算法包括 [1541] 提出的分布式剪枝协议（Distributed Pruning Protocol，DPP）、[329] 提出的三阶段一致阈值（Three-Phase Uniform Threshold，TPUT）算法和 [1774] 提出的一组带有 Bloom filter 优化的分布式阈值算法（distributed threshold algorithm，DTA）。

[1125] 提出了一族近似 top-k 查询处理算法，叫做 KLEE。KLEE 算法极大地减少了查询处理时的带宽消耗，只对排名前 k 的结果质量造成小的损失。每个端点维护了一个直方图，它对索引中的分数分布进行编码。直方图中每个单元保存一个概要：一个基于 Bloom

filter 的结构，表示分数落入该单元的文档集合中。这个数据在一个四步的近似查询执行算法中使用，实验表明，根据测试集的不同，它耗费的流量比 TPUT 最多减少一个数量级，同时仍保留大约 80%～90%的召回率。

[1048] 的方法建议通过广播来补充基于索引的查询处理，以避免在全局索引中维护大型记录列表。作者建议使用泛洪机制来回答热门的查询，而只有罕见的查询才依靠索引。有趣的是，这个方法与基于缓存的 P2P 系统相反。

[1464] 提出了一个用于关键词搜索的混合索引划分方案。所有端点聚成组，该索引方案在组内采用项划分，但是在组之间采用文档划分。因此，必须把每个查询广播到所有的组，但是在每个组内只有一些端点执行了实际的查询操作。因为组内的文档集大小是有限的，所以，与标准的 P2P 全局索引方法相比，这个方法降低了时延，高效地分散了带宽消耗。 443

尽管大多数方法采用端点级或文档级的索引颗粒度，但是 [1203] 提出了一个专注于平衡索引和查询处理代价的适应性方案。对于单个端点，本地文档组以项集的形式创建和展示，而项集通过索引来管理。因此，这样一个组级别的索引策略是这两种索引技术的泛化：端点级（每个端点一个组）和文档级（每个组一篇文档）。作者提出了一个概率模型来估计与给定数目的组相关的代价。

10.9 趋势和研究问题

对于解决与当前规模庞大的、不断增长的在线文档集相关的性能和扩展问题，并行计算有着很大的潜力。在本章中，我们调研了一些利用现代并行架构的技术。在并行硬件方面的趋势是通用 MIMD 机器的发展。与该趋势一致的是现代编程语言中出现的一些特性，如线程和相关的同步结构，它们极大地方便了在这些架构上的程序开发任务。尽管有这样的趋势，但在 MIMD 机器上的并行信息检索算法研究相对开始得较晚，只获取了很少的标准结果。

在并行信息检索方面的很多早期工作都是关注于在 SIMD 架构上支持签名文件。尽管 SIMD 机器很适合处理签名文件，但是 SIMD 机器和签名文件在它们各自的领域中都已失去"宠爱"。SIMD 机器很难进行编程，并且只适合相对较小的一类问题。如 9.3 节所指出的，签名文件对文档排序支持较差，只在很少一些方面胜过倒排文件，如功能性、索引大小和处理速度 [1799]。

分布式计算可以看成是 MIMD 计算的一种形式，但它有着相对较高的处理器间通信代价。然而，大多数在本章讨论的并行信息检索算法都有一个高的计算/通信比率，并且很适合对称多处理器和分布式实现。实际上，通过对处理器间通信使用合适的抽象层，我们可以方便地实现在多处理器和分布式架构上都能工作得很好的并行系统，并且只需要相对较少的修改。

在并行和分布式文本检索领域中仍然有很多挑战。尽管本章已经提出了一些方法，但还没有哪种方法能够脱颖而出，成为建立并行或分布式信息检索系统的确定方案。除了继续为基于倒排索引和后缀数组的系统开发和研究并行索引和搜索技术之外，有两个特定的挑战引人注目。

第一个挑战是在大型文档集上评价检索效果。尽管我们可以容易地评价给定并行系统所取得的加速比，但是评价系统结果质量就是另外一回事了。当然，这个挑战并不只是在并行信息检索系统中存在。大型文档集会造成一些问题，特别是在对查询生成相关性判断时。在文本检索会议（Text REtrieval Conference，TREC）中使用的聚合（pooling）技术可能也行不通。聚合技术合并来自多个系统的排序结果列表，产生相对较小的文档集，用于人工评价。它的假设是，即使不是全部，但大多数相关文档都包含在这个聚合中。对于大的文档集，这个假设可能并不成立。而且，并不清楚在这样的情况下召回率有多重要。 444

第二个重要的挑战是互操作性，或者在异质组件的基础上建立分布式信息检索系统。从 Web 上元搜索服务的流行性来看，对由异质的后端搜索服务器组成的分布式系统的需求是显而易见的。然而，因为缺少来自后端搜索服务器的项统计数据，所以这些系统的功能性受到限制，不能进行精确的重排序和结果列表合并。而且，每个搜索服务器采用定制的查询语言，当查询被翻译成后端查询语言时，它的原始目的就有可能丢失。STARTS[664] 等一些标准化协议尝试解决这些问题，但是需要整个搜索业界都遵照这些标准。

第三个挑战是用于设计大型信息检索系统的分析模型。Chowdhury 和 Pass[381] 介绍了一个基于排队论的方法，用来在吞吐量、响应时间和使用率等操作需求方面对搜索系统的架构进行建模和分析。最近，Badue 等人 [84] 提出了一个用于设计垂直搜索引擎的容量模型。

目前，建立分布式索引并不属于特别活跃的研究领域，也许是因为现有的技术已经在实际中取得了好的结果。但据我们所知，这些技术尚未在大型文档集上进行广泛测试，而验证结果是否满足预期是一个很重要的问题。

10.10 文献讨论

对并行和分布式计算的彻底的概览可以在《Parallel and Distributed Computing Handbook》(并行和分布式计算手册) [1800] 中找到，它由 Albert Zomaya 编辑。关于并行和分布式信息系统的很多有趣的研究论文可以在 IEEE《International Conference on Parallel and Distributed Information System》会议论文集上看到。

Stanfill 等人 [1528, 1530, 1531] 进行了很多早期的工作，使用大规模并行硬件 (特别是 Connection Machine) 来解决信息检索问题。Pogue 和 Willet[1287] 也尝试了大规模并行信息检索，使用的是 ICL 分布式阵列处理器 (Distributed Array Processor)。Salton 和 Buckley[1409] 对并行信息检索的早期实现提出了一些有趣的评论，质疑了它们的速度和有效性。

Lu 等人 [1059] 分析了如何恰当地扩展 SMP 硬件，用于并行信息检索，并且强调了恰当的硬件平衡的重要性。Tomasic 和 Garcia-Molina[1588, 1589, 1590]、Jeong 和 Omiecinski[833] 以及 Ribeiro-Neto 和 Barbosa[1349] 研究了并行和分布式倒排索引的实现技术。

Navarro 等人 [1183] 尝试了用于后缀数组构建和搜索的并行和分布式算法。对于 P 个处理器和总大小为 n 的文本，他们得到的平均索引时间包含了 $O(n/P\log n)$ 的 CPU 时间和
445 $O(n/P)$ 的通信时间。

Stanfill[1529]、Panagopoulos 和 Faloutsos[1239] 分析了 CM-2 (SIMD 架构) 中不同签名文件的性能。

Macleod 等人 [1072] 提供了一些用于构建联合信息检索系统的策略和建议。Cahoon 和 McKinley[309] 分析了 INQUERY 分布式信息检索系统的性能。

如下文献对源选择和文档集合并问题进行了研究：Gravano 等人使用的 GIOSS 系统 [667, 666]、Voorhees[1652]、Callan[323]、Moffat 和 Zobel[1152]、Viles 和 French [1640]，以及其他 [320, 437, 438, 515, 1053, 1300, 1473, 1475, 1730, 1760, 1797]。

在最近 7 年中，我们已经见证了不同 P2P 检索系统的快速发展，如 [462, 510, 1056, 1330, 1559, 1558, 1767, 1788, 1789]，它们的基本区别在于，在如何借鉴传统的信息检索技术，在 DHT 上设计一个更加强大的检索层。最近，有些更加具体的方向也得到了提高，包括 P2P 检索平台 [4]、索引算法 [1203]，以及十分重要的如何分析和比较不同的方
446 法 [1198, 421]。关于 P2P 检索比较好的综述是 [1766, 1358]。

Web 检索

——与 Yoelle Maarek 合著

11.1 介绍

Tim Berners-Lee 在 1989 年提出了概念性 Web，然后在 1990 年 12 月成功地验证了 Web[192]，并在 1991 年年初发布了第一个 Web 服务器。它称为万维网（World Wide Web），但它在本书中被简单地称为 **Web**。在那时，没有人能够想象到 Web 的影响。在海量数据和信息呈指数级增长的驱动下，Web 不断地发展，各种各样的日常任务，如电子商务、银行、研究、娱乐，以及个人通信都无法在 Web 之外便捷和低代价地完成。

Web 中可用的文本数据的数量据估计是 PB 级的。另外，图像、音频和视频等其他媒体也是海量的，规模甚至更大。这样，Web 可以被看成一个非常大的、公开的、非结构化的，但却无所不在的数据库，这就需要有效的 Web 信息管理、检索以及过滤的工具。所以，Web 搜索引擎成为了互联网中最常用的工具之一。另外，信息发现在大规模企业网[⊖]中也变得更为重要，用户可能需要抽取或推断出新的信息来支持某个决策过程，这个任务称为数据挖掘（或针对 Web 的特定情况，称为 Web 挖掘）。

大规模的可用数据加上快节奏的变化，使得从 Web 上搜索相关信息变得非常困难。为了应付快节奏的变化，高效的网络爬取变得非常重要（见第 12 章）。另外，爬取和其他任务也是相关的，如信息抽取和 Web 数据挖掘。 447

尽管总体来说最近在图像以及非文本数据搜索中取得了一些进步，但现存的技术不能很好地应用在 Web 上（见第 14 章）。因此，在文本上的搜索一直是最流行的热门研究课题。大多数搜索引擎位于美国，关注于英文文档，但是还存在一些重要的非美国的搜索引擎，为特定的语言而设计，可以处理各种文字体系和字母表，如汉字或西里尔字母。这些搜索引擎的例子包括中国的百度、俄国的 Yandex 和韩国的 Naver。

我们也注意到搜索继续被“句法”（syntactic）范式主导着，包含用户指定词语或模式的文档会被检索出来。如第 3 章讨论的，这种词语或模式不一定能反映文本的内在语义。句法搜索的一种替代方法是对文本进行自然语言分析。自然语言的前期处理和抽取文本语义的技术已经出现一段时间了，但它们还不是非常有效。实际上，除了最近提出的一些快速的实体抽取工具外 [79]，这些技术代价很高以至于没办法应用于大规模数据 [86]。另外，在大多数情况下，它们只对良好书写和结构化的文本有效 [1765]，并需要结合同义词典或者其他上下文信息，如特定的语言领域。对于结构化数据检索更加详细的讨论见第 13 章。

为了简化这个问题，搜索引擎的设计者做出了关于 Web 的若干假设，但它们并不总是成立的。关于数据，他们隐含假设一个包含网页或者另一种数据的物理文件是一个单一的逻辑文档。然而，每个网页可能有多个逻辑文档（例如一份报纸）或者一个文档包含很多文件（例如一篇学位论文）。至于用户的需要，他们最初假设用户的主要目标是直接的信息搜寻，而且这些需求的多样性比他们真正的需求更小。然而，就像我们在第 7 章讨论的那样，这两

⊖ 企业网指在组织内部建立的计算机网络，可以与或不与因特网相连。

个假设不再成立。

探索 Web 基本上有两种主要的形式。第一种是向索引了部分 Web 文档的搜索引擎提出一个基于词的查询。第二种是浏览 Web，它可以看成是跟随超链接的连续搜索过程，例如在将 Web 文档按照主题分类的 Web 目录中。还存在着其他的方法，例如利用 Web 的超链接㊀结构，尽管它们还不是全部可用，可能不是很有名，也可能更复杂得多。我们在这里涉及 Web 搜索的全部形式，更强调前两种。

本章按如下方式组织。首先讨论搜索 Web 的挑战，然后通过介绍 Web 的特点，方便读者更好地理解问题的复杂性。接下来详细讨论搜索引擎的架构、排序模型，以及 Web 数据管理。然后充分地探讨了用户一般如何与搜索引擎交互的相关问题。接下来介绍浏览，将其与搜索做比较，还讨论浏览之外的方法。最后对一些和 Web 搜索相关的问题和挑战进行总结，例如 Web 挖掘、计算广告、元搜索、目前的趋势和研究问题。因为 Web 研究是一个非
448 常动态的领域，我们可能漏掉了一些重要的工作，在这里提前向读者道歉。

读者应该意识到本书付印时，或者从书架上被挑中时，以及在随后几年阅读时，书中描述的很多特征将会过时㊁，引用的很多网址有可能变成破碎的链接，或者它们的内容已经改变。在极端的情况下，有些主要的搜索引擎完全消失了㊂，或者一些新的已经出现。即使这样，我们也希望这里描述的基本原则在未来几年都能保持稳定，为有兴趣的读者提供搜索引擎背后的基本科学知识。

11.2 一个有挑战性的问题

现在让我们考虑 Web 搜索提出的主要挑战。我们可以把它们划分为两类：和数据本身相关的问题，我们称为以数据为中心的问题；以及用户和用户与数据交互的问题，我们称为以交互为中心的问题。以数据为中心的问题多种多样，包括：

- **分布式数据** 由于 Web 的内在本质，数据跨越大量的计算机和平台。这些计算机都是相互连接的，没有预定义的拓扑结构，另外在带宽可用性和网络互联的可靠性方面也千差万别。
- **高比例的不稳定数据** 由于互联网的动态性，新的计算机和数据很容易增加或删减。例如，早期的估计表明 50% 的 Web 在几个月中发生变化 [859, 793, 257, 373, 561, 1215, 495]。搜索引擎也面临着悬垂的（或破碎的）链接，以及当域或文件名变化或消失时的重定位问题。
- **大容量的数据** Web 快速发展带来的扩展性问题很难解决，另外在实践中动态网页是无限的。
- **非结构化和冗余的数据** Web 并非如有些人认为的那样，是一个巨大的分布式超文本系统，因为它不遵循严格的内在基本概念模型以保证其一致性。事实上，Web 无论在全局上还是在个体的 HTML 页面级别上，都没有很好地结构化。在最好的情况下，HTML 页面被认为是半结构化数据。此外，大量的 Web 数据或者松散地（如同一个新闻机构创建的新闻），或者严格地（通过镜像或复制）重复。大约 30% 的网页是（近似）重复的 [269, 1467, 278, 559, 560, 120]。语义冗余可能就更多了。
- **数据的质量** Web 可以被视为一个新兴的出版媒体。然而，在大多数情况下，却没

㊀ 我们用超链接或链接代表从一个网页到另一个网页的指针（锚）。

㊁ 见之前的关于 Web 快速发展的评论。

㊂ 考虑当本书第 1 版出版后 Excite 引擎的情况。

有编辑的过程。所以，数据可能是不准确的、完全错误的、过时的、无效的、书写不好的；或者在许多情况下，充满了错误，无论是无意的（错别字、语法错误，OCR 错误）还是有意的。错别字和错误，尤其是外国人名字的错误是非常常见的。 449

- **异质数据** 数据不仅来源于各种媒体类型，每种类型来自不同的形式，而且它也通过不同语言进行表达，这些语言有各种字母表和文字体系（如印度），字母表也可能会相当大（例如汉语或日本汉字）。

很多挑战，如数据类型的多样性以及较差的数据质量，都无法通过制定更好的算法和软件得到解决，这将长期成为事实，因为它们是人性所固有的困难和问题（例如语言的多样性）。

第二类挑战是当用户和搜索系统交互时所面对的问题。以交互为中心的问题包括两种：

- **表达查询** 人们的需求或者所要完成的任务通常都不容易以“查询”的形式表达。即使以更自然的方式表达查询，也只是信息需求的一种反映，因此，按照定义，它是不完善的。这种现象可以比做“柏拉图的洞穴”，那里的影子被误认为现实。
- **解释结果** 即便用户能够完美地表达查询，但答案[⊖]可能也会被拆成几千甚至几百万的网页，或者根本不存在。在这种情况下，有很多问题需要解决。例如，如何处理很大的答案集？如何对结果排序？如何选择用户真正感兴趣的文档？甚至在只有一个候选文档的情况下，文档本身可能非常大，如何高效地浏览文档？

对于 Web 所带来的内在问题，用户最主要的挑战是想出好的查询提交给搜索系统，它将生成可管理的、相关的答案。搜索系统主要的挑战是快速地搜索和给出相关的答案，即使对于表示得较差的查询，因为这是 Web 中常见的情况（见 7.2.1 节）。

在 Web 当前的状态下，搜索引擎需要处理纯 HTML 和文本，以及其他数据类型，如多媒体对象、XML 数据和相关的语义信息，这些数据能够动态地产生，本质上更复杂。为了将这个区别描述得更清晰，在本章其余的部分，我们用“*网页*”来表示 HTML 文档（见 6.4.2 节），并用“*Web 文档*”来表示 Web 中所有可用的数据类型。

如果语义网克服了它所有的内在的社会化问题，成为了现实，那么一个基于 XML 的、带有标准的语义元数据和模式的 Web 也可能会成为现实。在这个假想的世界中，信息检索变得更简单，甚至多媒体搜索可以被简化。垃圾可以更容易地消除，因为可以更容易地识别出好内容。另一方面，新的搜索问题可能会出现，例如极大规模的 XML 处理和检索以及在结构化数据上的 Web 挖掘。 450

11.3 Web

在本节中，我们讨论 Web 的主要特点，以及描述其内容和结构的数学模型。已经有很多研究调查了某些特定国家的 Web，研究表明很多 Web 子集的属性和特点在全球 Web 的范围内也是有效的（且适用的）[97]。尽管这样，我们仍然要对 Web 及其动态性有充分的理解 [794，258，1361，98，1011，1010，890]。

11.3.1 特性

由于其高度的动态性，评价 Web 是一个困难的任务。在本书写作的时候（2010 年 4 月

⊖ 有时我们会用“答案”来表示给定查询的答案集合。上下文会清晰地指示，对于自然语言的问题，我们考虑的到底是一个答案集合还是一个特定的答案。

的 ISC 调查)，有超过 200 个国家的 7.5 亿台计算机直接连接到因特网上，它们中许多都是 Web 服务器 [812]。此外，根据 Netcraft 网站的调查，Web 服务器的估计数量目前已经超过 2.06 亿 [1196]。根据这些数字，我们可以说，每 4 台直接连接到因特网的计算机中，就有一台是 Web 服务器。

在两篇有趣却已过时的文章中，Bray [250] 和 woodruff 等人 [1719] 研究了早期 Web 的不同统计指标。根据第一篇文章，在 1995 年 11 月有 1100 万页面，而从第二篇得到的数据是 260 万网页。第一个问题是有多少不同的机构（而不是 Web 服务器）维护 Web 数据。这个数字小于服务器的数量，因为很多地方有多个服务器。确切的数字是未知的，但应该大于 Web 服务器数量的 40%（这一比例是 1995 年的数据)。1998 年年初的规模估计是2～3.2 亿，1998 年 7 月的最佳估计达到 3.5 亿 [198]。更多关于搜索引擎规模的近期研究 [135, 685] 估计 2005 年有超过 200 亿页面。静态网页[㊀]的大小大约每 8 个月翻一倍。在动态网页[㊁]变得流行之前，静态网页的确切数目是非常重要的。如今，Web 对于实际用途来说是无限的，因为它可以产生无限的动态页面（如一个在线日历)。12.3.1 节将介绍页面的一个分类体系。

最流行的 Web 文档的格式是 HTML，之后依次是 GIF 和 JPG（都是图像)、ASCII 文本和 PDF [97]。最流行的压缩工具是 GNU zip、Zip 和 Compress。关于 HTML 页面，有很多有趣的特点和统计数据。第一，大多数 HTML 页面不是标准的，这意味着它们没有遵循全部的 HTML 规范。事实上，如果浏览器是严格的 HTML 编译器，很多页面不会被渲染。另外，尽管 HTML 是 SGML 的一个实例，但 HTML 文档很少以正式的文档类型定义开始。第二，HTML 页面是很小的（大概 10KB)，通常包含很少的图像。大多数页面为了展示目的而使用图像，如彩色的锥体和线条。每个页面平均包含 5～15 个超链接（平均多于
451 8 个链接）它们中的大多数是本地的，也就是说，它们指向自己的 Web 服务器层次结构中的网页。平均来看，指向任何一个给定页面的外部页面的数量接近于零。典型地，指向给定页面的只有本地链接，也就是来自相同域的页面，甚至对于大型网站的主页来说也是正确的。

被引用最多[㊂]的网站是主要的互联网公司。另一方面，有很多链接到外部网站的网站目录，如雅虎[㊃]目录或开放目录计划（Open Directory Project)[㊄]，以及维基百科[㊅]等 Web 2.0 站点。在某种程度上，这些聚合链接的站点是 Web 的“胶水”。没有它们，我们会有更多孤立的部分或“岛屿”，就如同许多个人网页的情况。

对于在网页中使用的语言，有三项早期的研究。第一项研究是 Funredes [1263] 在 1996—1998 年所做的，后来继续进行到 2005 年。它应用 Alta Vista 搜索引擎，搜索不同语言的不同词语。后来，也应用了其他搜索引擎。这种技术在统计上可能不是很显著的，但是其结果和第二项研究是一致的。这项研究是 Alis Technology[28] 所做的，它基于一个自动软件来检测所用的语言。这项研究的一个目标是通过在 8000 个 Web 服务器上运行并测试该软件。第三项早期研究是 OCLC 在 1998 年 6 月 [1230] 做的，它通过对因特网 IP 地址进行抽样，并应用 SILC 语言识别软件完成。由于目前 Web 的巨大规模，还没有

㊀ 网页是 Web 服务器上的文件。
㊁ 动态网页是用户和 Web 服务器交互时建立的 HTML 页面。
㊂ 我们谈论的是被引用至少 100 万次的页面。
㊃ Yahoo. com/dir。
㊄ dmoz. org。
㊅ wikipedia. org。

更新的研究。

2000 年，估计约有 70%的网页是英文的，但其他语言可用词语的数量比英语词语增长得更快［676］。2003 年 1 月，谷歌 Zeitgeist⊖显示谷歌中的英语查询从 2001 年的约 60%降到约 50%。更新的结果只是适用于特定的国家［97］。

11.3.2 Web 图的结构

将 Web 视为一个图，现在已经被广泛地认可。从最简单的视角来看，结点代表单个页面，边代表页面之间的链接。Web 图的全局结构已经被广泛地研究。Broder 等人在［271］中做了最完备的研究，将 Web 图的拓扑结构和蝴蝶结结构（bow-tie）进行比较。这个比喻在图 11-1 中粗略地表示。发挥一些想象，读者可以看到一个蝴蝶结，其中最大的强连通分支（Strongly Connected Component，SCC）扮演了蝴蝶结的中心结的角色。

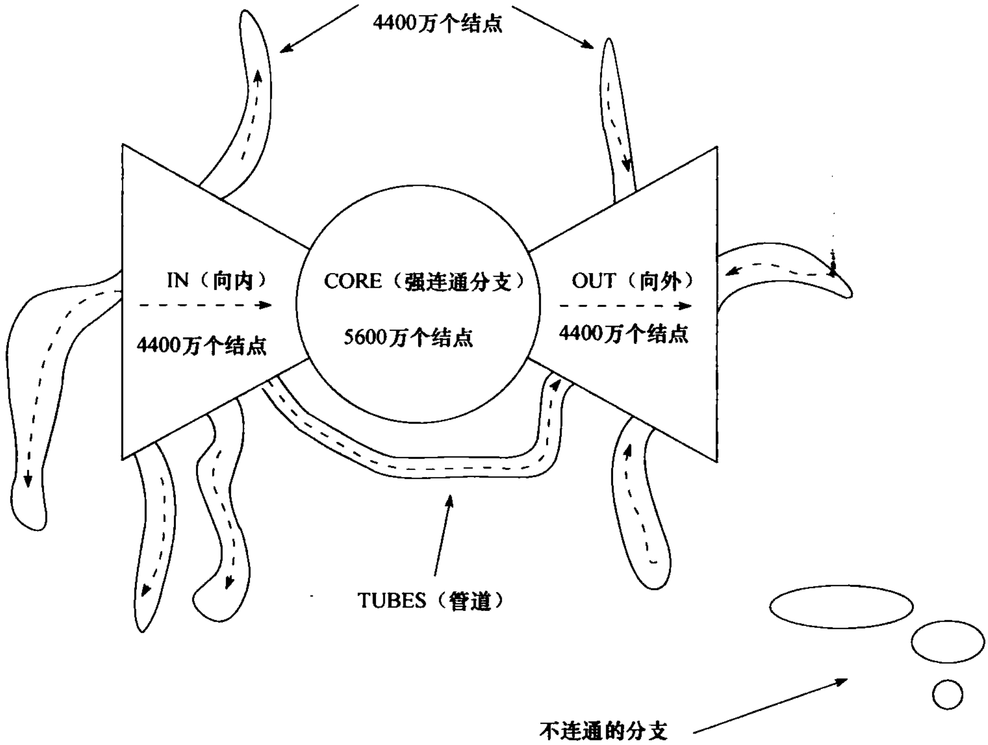


图 11-1 Web 的原始"蝴蝶结"结构（改编自［271］）

这种模型的进一步完善指出了 SCC 或 CORE 分支内部的区域，如［98，506］所描述的。正如我们在 11.1 节中提到的，这种模型的一个限制表现在，一个网页并不总是一个逻辑单元。图中被识别出的分支如下：

- CORE（核心）：组成图中强连通分支的站点。通过定义，人们可以从 CORE 中的任何站点导航到 CORE 中的其他站点。 452
- IN（向内）：可以到达 CORE 中的站点，但是不能从 CORE 中的站点到达。
- OUT（向外）：可以从 CORE 中的站点到达，但是没有回到 CORE 的路径。

⊖ URL：http://www.google.com/press/zeitgeist.html。

- TUBES（管道）：在 CORE 外面直接连接 IN 和 OUT 的路径上的站点。
- TENTACLES（触手）或者 TENDRILS（卷须）：能够从 IN（T. IN）中的站点到达的站点，以及只能到达 OUT（T. OUT）、但是不属于之前分支的站点。
- DISCONNECTED（不相连）或者 ISLANDS（岛屿）：不相连的站点，它们连接的分支可能和整个 Web 有相似的结构。

在 [94] 中，这个表示通过将 CORE 分支分成四个部分来扩展，如下所述。

453
- Bridges（桥）：CORE 中可以直接从 IN 分支到达，同时又能直接到达 OUT 分支的站点。
- Entry points（入口点）：CORE 中可以从 IN 分支直接到达但不在 Bridges（桥）中的站点。
- Exit Points（出口点）：CORE 中可以直接到达 OUT 分支但不在 Bridges（桥）中的站点。
- Normal（标准）：CORE 中不属于之前定义的子分支。

图 11-2 显示了更加完善的"蝴蝶结"结构。

在所有关于 Web 的（有限制的）研究中 [97]，CORE 分支由较少数的网站组成。另一方面，它有很高的网页密度。结构和内容的质量也显示有一些相互的关系，这些关系已经从链接分析中得到（连接度越高，则质量越高）。有些研究表明 ISLANDS 的数量比我们认为的更多，它们中的大多数不与 Web 相连，因此，除非它们在搜索引擎上注册，否则很难被发现 [121]。

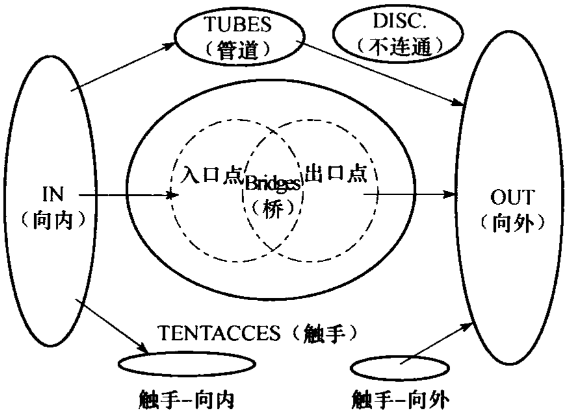


图 11-2 Web 宏观"蝴蝶结"结构的缩略图描述

11.3.3 对 Web 建模

Web 的所有特点显示 CORE 分支的定量属性服从幂律（power law）分布。一个普遍的幂律是一个对于尺度变化保持不变的函数。其最简单的形式是：

$$f(x) = \frac{a}{x^{\alpha}} \quad \alpha > 0 \tag{11-1}$$

很容易验证，cx 形式的尺度变化，只会改变常数 a 而不会改变函数的形态，其中 c 是常数。这种不变性通常也称为自相似性。如果 $f(x)$ 是一个概率分布，则一定要设置 a 的值，使得所有的概率之和是 1。那么只有当 $\alpha>1$ 时，a 才作为一个常数存在。在那种情况下，依据 α 值的不同，分布的矩可能是有限的或无限的。当 $\alpha \leqslant 2$ 时，均值和所有的高阶矩是无限的；当 $2<\alpha \leqslant 3$ 时，均值存在，但是方差和高阶矩是无限的。遵循幂律的 Web 测度

454 的例子包括：

- 每个网站的网页个数和每个域的网站个数，就像内容研究所显示的。
- 入链和出链的分布，以及连通分支的个数，就像链接结构研究所显示的。

这不仅在 Web 图中，而且在由网站级连接所构成的主机图上，都是正确的。表 11-1 总结了 [97] 的主要发现。对于页面的大小，有两个幂指数：一个针对小于 20KB 的页面，一个针对其他。出度也是相同的：一个针对出链小于 20 的页面，一个针对更多出链的页面。

表 11-1 各个国家和地区 Web 的幂指数的总结

地区	页面大小		每个站点的页面数	入度	出度	
	小	大			小	大
巴西	0.3	3.4	1.6	1.89	0.67	2.71
智利	0.4	3.2	1.6	2.01	0.72	2.56
希腊	0.4	3.2	1.6	1.88	0.61	1.92
印度支那	n/a	n/a	1.2	1.63	0.66	2.62
意大利	n/a	n/a	1.3	1.76	0.68	2.52
韩国	0.4	3.7	3.2	1.90	0.29	1.97
西班牙	n/a	2.25	1.1	2.07	0.86	4.15
英国	n/a	n/a	1.3	1.77	0.65	3.61
全世界	n/a	n/a	n/a	2.1	n/a	2.7

我们这里主张，以和 6.5 节相同的方式对整个 Web 文档的特性建模是可能的。基本上，Heaps 和 Zipf 法则在 Web 尺度上是合理的，且有更快速增长的词汇（较大的 β）和更偏置的词语分布（较大的 α）。

另外，文档大小的分布也可以通过一个数学模型来描述，该数学模型将文档大小看成自相似的 [458]，也就是说，它们对于尺度变化是不变的（相似的行为出现在 Web 流量上）。最佳模型是基于两种不同分布的混合模型。分布的主体服从对数正态分布，大小为 x 字节的文档的出现概率服从以下分布：

$$p(x)=\frac{1}{x\sigma\sqrt{2\pi}}e^{-(\ln x-\mu)^2/2\sigma^2} \tag{11-2}$$

其中均值（μ）和标准差（σ）分别是 9.357 和 1.318 [146]。图 11-3a 显示了在一个抽样文档集中文件大小的分布，其中所有的对数是以 10 为底的。注意尾部作为负斜率的直线，出现在水平轴值 4 的右侧。

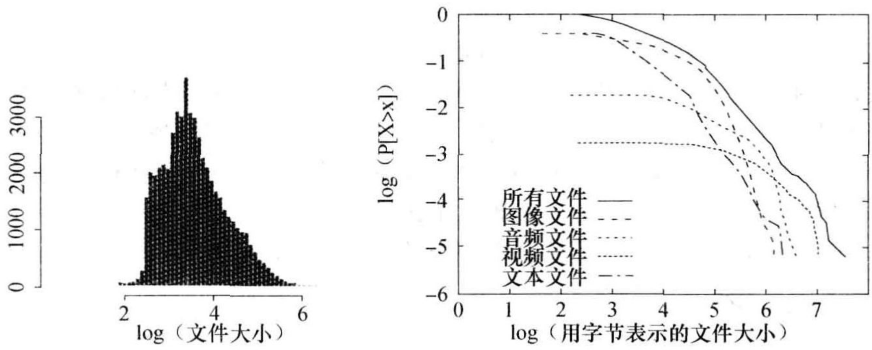


图 11-3 a）用半对数图表示的所有文件大小的分布（摘自 M. Crovella，1998）；b）在对数-对数图中，不同类型文件大小的概率分布的尾部 [458]

图 11-3b 的尾部是“重尾”。也就是说，大部分文档都很小，但是也有不少大文档。对于图像或视频文件这是很直观的，在 HTML 页面上这也是正确的。通过帕累托分布（另一个幂律函数）能得到一个不错的拟合，即 455

$$p(x)=\frac{\alpha k^{\alpha}}{x^{1+\alpha}} \tag{11-3}$$

其中 x 用字节来衡量，k 和 α 是分布的参数 [146]（见图 11-3b）。长尾体现在，曲线在右侧下降为负斜率的直线。对于文本文件，α 的值在 1.36 左右；对于图像和其他二进制格式，这个值会更小 [458，1699]。考虑到所有的 Web 文档，我们取 $\alpha=1.1$，$k=9.3$KB。对数正态分布和帕累托分布之间的切割点大于 9.3KB（也就是说，在水平刻度 3.98 之后，它使得

整体概率累加为 1)。有 93%的文档，其大小小于切割点。实际上，对于小于 50KB 的文档，典型的文件是图像；50～300KB，声音文件的数量会增加；从 300KB 增加到数兆字节，视频文件更常见。这些分布的参数从 2 个月间多个用户的 46 000 多个网页请求的抽样中得到。相关信息能够在 Web 基准上找到，如 WebSpec96 和 Sun/Inktomi Inkbench[811]。

11.3.4 链接分析

在 Web 上，遵照 [92]，我们将链接分析分为 3 个等级：

- 微观级 (microscopic level)，与链接和单个结点的统计属性相关。
- 链接分析的介观级 (mesoscopic level)，与 Web 的地区属性相关。
- 链接分析的宏观级 (macroscopic level)，与大型 Web 的结构相关。

Web 的宏观级描述开始于 Broder 等 [271] 关于“蝴蝶结”的重要工作，我们已经在
456 11.3.2 节进行了讨论。一个相关的宏观描述是 [1565] 所提出的 Jellyfish 结构，它可以被应用到互联网的自治系统中。根据这个观点，我们可以识别被链接密度不断下降的区域所包围的核心部分，它带有很多结点，形成长且松散相连的链或触手 (tentacle)。

介观级的链接分析和结点邻域的属性有关，这是大多数基于链接的排序函数的工作环境。一种描述结点邻域的方式称为“hop 图”：它是一种表示在不同距离下的不同领域个数的图，如图 11-4 所描述的。介观级也是可以观察到局部结构描述的级别，这些局部信息包括结点的社区信息和聚类等。

Web 的微观级描述已经被多名作者讨论 [792, 139]，他们的观察是，基于网页链接数的分布是倾斜的 (skewed)，而不是在经典随机图中所观察到的典型泊松分布 [536]。在无尺度网络 (scale-free network)，例如 Web 中，网页 p 的链接数的分布服从幂律：

$$Pr(p\text{ 有 }k\text{ 个链接}) \propto k^{-\alpha} \qquad (11\text{-}4)$$

其中，通常有 $2<\alpha<3$，这暗示了一个有限的均值 (见 11.3.3 节)。

无尺度网络有一些强连通的链接，它们作为“枢纽”连接网络中的很多其他结点。无尺度网络的连通性对于边的随机去除是稳定的 [324]，可以部分解释成“优先连接” (preferential attachment) 过程 [140]，也叫做富者益富现象或 Yule 过程。在这个过程中，新网页 v 链接另一个网页 w 的概率和 w 的入链数成正比。

图 11-5 显示了我们所介绍的链接分析等级的一个可视化的描述。

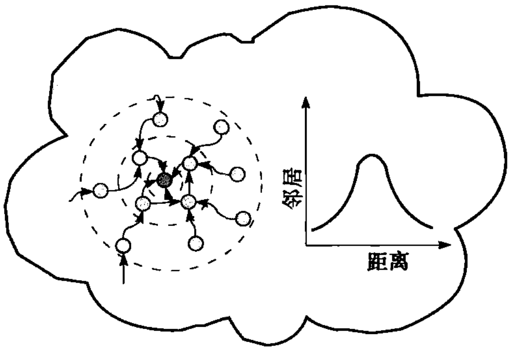


图 11-4 “hop 图”的示意图描述：给出了在不同距离的邻居数量 [92]

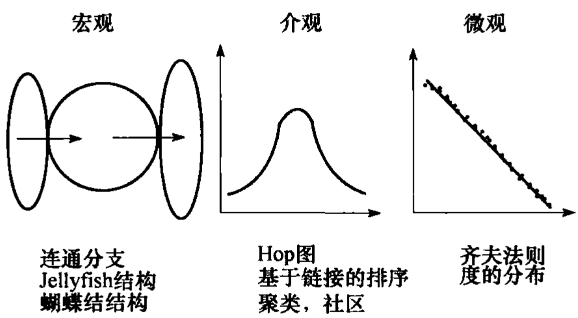


图 11-5 链接分析的级别 [92]

链接分析是一个极其丰富的信息源，不仅可以用做推断相关性 (将在 11.5.2 节讨论)，还可以用来确定爬取的优先度 (见 12.5.1 节)，甚至可以在 Web 图中找到社区等子结构

[625，947]。在 11.5.2 节，我们应用链接分析，利用链接来定义不同的排序方式。 457

11.4　搜索引擎架构

在本节中，我们将讨论检索系统的不同架构，它们将 Web 建模成一个全文数据资源库。标准的信息检索系统和 Web 搜索的一个主要差异是，在 Web 上，所有的查询处理和排序都必须是在不访问源文档的情况下完成的。这就避免了在查询期间访问远程网页的缓慢过程。这个基本差异对于索引和搜索算法，以及查询语言的复杂性都有直接的影响。需要注意的是，为了产生片段（snippet），也会使用源文档，但这仅限于前 10 个要展示给用户的结果。

11.4.1　基本架构

大多数搜索引擎采用*集中式爬取器-索引器结构*（centralized crawler-indexer architecture），如第 12 章讨论的。爬取器，或称爬虫，是一种程序（软件代理），它遍历 Web，发送新的或者已更新的页面给负责索引的主服务器。爬虫也称为机器人（robot）、蜘蛛（spider）、漫游者（wanderer）、步行者（walker）和智能机器人（knowbot）。尽管它叫爬虫这个名字，但爬虫实际上并没有被发送到远程机器上执行。相反，它在本地系统上运行，并发送请求到远程的 Web 服务器。

索引以集中的方式回答从 Web 不同地方提交的查询。大多数搜索引擎应用倒排索引的变体（见 9.2 节）。简单地说，倒排索引由索引项列表（词汇表）组成，其中每个索引项与出现它的页面的指针列表相关联。需要记住的重要一点是，只对文本的逻辑视图建立索引，而不是文字视图。事实上，归一化操作会按常规进行，可能包含去除标点，去除词对之间多余的空格，大写字母转换为小写字母（见第 6 章）。有些搜索引擎通过去除禁用词来减少索引的大小。然而，由于它们必须处理数百种语言，因此禁用词是通过统计来选择的。 458

为了使结果更丰富，使用户可以对结果页面上的每一个答案有所了解，索引会补充与每个网页相关的元数据，如它的创建时间、大小、标题等。假设存储每页的 URL 和元数据需要 500 字节，那么为 10 亿页面存储信息一共需要 500GB。这些信息可以有效地压缩，所以在实际中，其真实大小会显著减小。

给定一个查询，显示的答案集合是完整集合的一个子集（通常是 10 个结果）。如果用户需要更多的结果，那么搜索引擎能够重新计算这个查询来生成后面的 10 个结果，或者从保存在主存内的部分结果集中得到它们。在任何情况下，搜索引擎从来不会从整个 Web 文档集中计算全部答案，因为找到数千个最相关的结果通常是足够的。事实上，计算完整的答案集会非常慢，因为有些查询有很多的结果。

最先进的索引技术能够将倒排索引减少到原始大小的 30%（如果去掉禁用词，甚至会更少），如 9.2.6 节讨论的（见 [1703]）。为了说明这点，考虑到 10 亿页面的 Web 文档集需要 1.5TB 的存储空间。未压缩的索引需要大约 60%的空间，即 900GB。另一方面，压缩索引大约需要不到一半的空间，也就是 400GB。正如 9.2.3 节讨论的，通过结合单个词的文档列表来产生最终列表，索引可以用于回答由多个词组成的查询。很多搜索引擎也支持精确短语和邻近搜索，这些功能或者需要文档中查询项位置的附加信息，或者需要将频繁短语作为索引单元（或者二者兼具）。

如果每一个词都不是很频繁的，那么搜索步骤就可以有效地进行。然而，在 Web 中很少是这样的。为此，当潜在答案的数目可能非常大时，所有的搜索引擎用一种懒惰的查询处理方案。也就是说，只计算第一个答案，当用户在看过第一个结果以后再请求进一步的结果

时，才会计算进一步的结果。更多通过倒排索引搜索的细节可以在 9.2.3 节看到。

图 11-6 显示了 Alta Vista 等早期搜索引擎［426］的软件架构示意图。它由两部分组成：一个处理用户的请求，包括用户界面和查询引擎组件；另一个处理数据，包括爬取和索引组件。1998 年，整个系统在 20 台多处理器的机器上运行。它们全部加起来有超过 130GB 的 RAM 和超过 500GB 的磁盘。查询引擎本身使用超过 75％的资源。

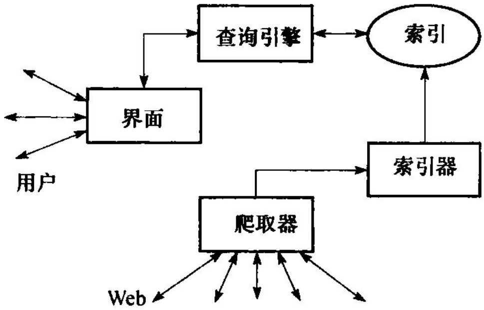


图 11-6 早期爬取器-索引器的典型架构

这个架构所面对的主要问题，除了收集数据，就是它的容量。实际上，爬取器-索引器架构在 20 世纪 90 年代末就无法应对 Web 的增长。解决方法是分布式和并行的计算，正如下面所解释的。

11.4.2 基于集群的架构

目前的搜索引擎使用大规模并行和基于集群的架构［151，484，99］（见第 10 章）。由于文档
459
集较大，倒排索引无法存放在单台计算机上，必须分布到集群中的多台计算机上。为此，如 10.3 节所述，应用文档分割技术。大量的查询表明这种基本的架构必须被复制多份以便能够处理整体的查询负载，而每个集群必须处理查询负载的一部分。另外，因为查询来自世界各地，跨各大洲有明显的互联网时延，集群副本应保存在不同的地理位置，以减少应答时间。这使得搜索引擎在大多数典型的最坏情况下可以容错，例如停电或自然灾害。在这种架构中有很多重要的细节需要仔细地分析：

1）在搜索引擎的内部（回答查询和索引）和外部（爬取）之间实现很好的平衡是尤其重要的。这是通过分配专用集群给爬取、文档服务、索引、用户交互、查询处理，甚至是结果页面的生成来完成的。

2）另外，也需要维持不同集群间的良好负载平衡。这是通过称为负载均衡器（load balancer）的专门服务器完成的。

3）最后，因为硬件经常中断，因此容错需在软件级进行处理。将查询分发到最胜任的集群上，同时，使用廉价的可替换的硬件部件，在发生故障时处理器和硬盘可以替换。

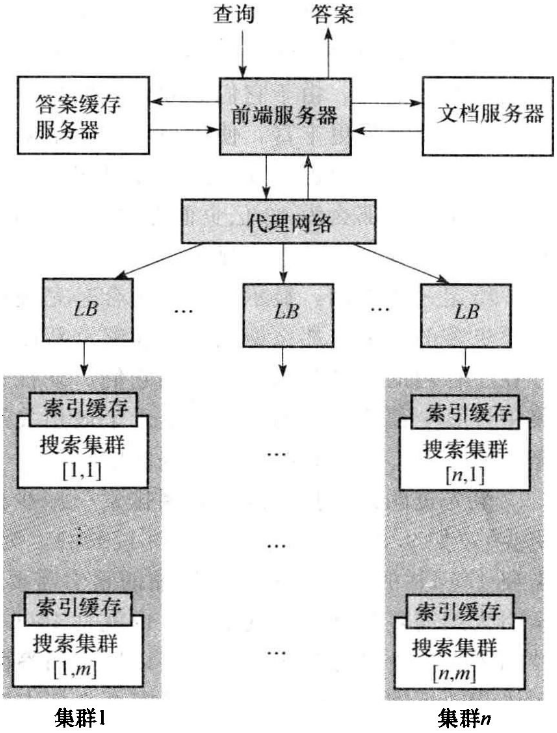


图 11-7 基于集群的架构中的搜索模块［484］。每个集群包括一个所有文档集的索引，该索引在集群中的 m 个服务器间被分割。利用 n 个集群来生成整个索引的 n 个副本

图 11-7 显示一个通用的具有关键组件的搜索集群架构。前端服务器接收查询，如果答案已经在“答案缓存”服务器中，那么系统就马上处理它们（见 11.4.3 节）。否则，它们通过层次式代理网络将查询发送给搜索

集群。这个网络的确切拓扑结构会有所不同，但基本上其设计应该可以平衡流量，以尽可能快速地到达搜索集群。每一个搜索集群包括一个负载平衡服务器（图 11-7 中的 LB）将查询发送给搜索集群的全部服务器。在图 11-7 中，我们展示了将索引分配到 m 个服务器上，整 460
个集群则有 n 个索引副本。尽管将索引分配到单个集群上是可以的，但并不建议这么做，因为集群可能会变得非常大，接下来会遭受额外的管理和容错问题。每个搜索集群也包括一个索引缓存，如顶部的平面矩形所示。代理网络将搜索集群的结果融合起来，把融合的结果发送到适合的前端服务器，这些服务器可以使用正确的文档服务器产生完整的结果页面，其中包括摘要和其他搜索结果页面。这是更普遍的、将整个数据中心看成一台计算机的趋势的一个例子 [152]。Orlando 等人 [1235] 提出了一个并行和分布式搜索引擎架构，它基于两个主要的并行化策略：一个任务并行策略（多个同质的索引服务器独立地执行查询）和一个数据平行策略（访问数据库不同分区的索引服务器并行处理查询）。然而，实际中很少公布这种架构的细节描述，因为领先的搜索引擎将这些细节（大多数查询过程所需资源的确切数字，如 CPU 和磁盘）作为商业秘密。 461

Chowdhury 和 Pass[381] 介绍了文档划分架构的排队论模型，并应用它来分析内在的运行要求：吞吐量、响应时间和利用率。他们的排队模型假定对于每个查询，处理相同数量文档的索引服务器的执行时间之间有一个完美的平衡。然而，Badue 等人 [85] 发现即使索引服务器间的文档集有平衡的分布，但查询日志里查询项的频度和相关倒排表间的联系也可以导致在这些服务器中查询执行时间的不平衡，因为这个联系影响磁盘的高速缓存行为。此外，每台服务器主存的相对大小（和磁盘的使用情况有关）和参与到并行查询处理任务的服务器数量也能够造成局部查询执行时间的不平衡。通过用于该特定架构的容量规划模型，同样的作者 [84] 解决了这个问题。尽管如此，关于 Web 搜索系统性能模型的可用文献仍然是有限的。

Web 搜索引擎的规模和复杂性，以及每天用户提交的查询量，使查询日志成为提高搜索结果的精度和搜索引擎中各个部分效率的重要信息源。查询（项）分布、每个查询的到达时间和点击结果等特征是从查询日志上抽取信息的一些例子。现在考虑的重要问题是，我们是否可以利用和转换这些信息，以便对文档集进行划分，更高效和有效地分发查询。在过去的几年中，很多研究关注使用查询日志进行查询缓存，对文档集进行划分和执行查询分发 [1304，1474]。下一步，我们探索一些对理解查询负载和搜索引擎的查询分布有益的技术。

11.4.3 缓存

因为搜索引擎需要快速工作，所以无论何时大多数任务被认为应该在主存中进行。因此，缓存被大力推荐和广泛使用。缓存对于被大量用户访问的 Web 系统来说是一个有用的技术。它能够减少平均响应时间，显著地减少后端服务器的工作量，并减少带宽的使用总量。在 Web 系统中，客户端和服务器都可以缓存对象。虽然浏览器和代理可以缓存客户端上的 Web 对象，但服务器缓存预先计算的结果或新结果计算中使用的部分结果 [1284]。我们专注于搜索引擎端缓存的讨论，因为这是更有效的并且完全受到控制的。这种缓存机制受用户共享某些相同查询这一现象的驱动，如最初在 [1727] 中所记录的那样。

搜索引擎中最有效的缓存技术是搜索结果的缓存或叫*答案缓存*，它是在图 11-7 中前端层进行的，它允许对频繁出现的查询进行快速响应。因为查询服从幂律分布，少量的查询经常重复，因此一个小的缓存可以回答大比例的查询。例如，如果我们得到 30% 的命中率，那么搜索引擎的容量就能增加近 43%。另一方面，在任何时间窗口内，大部分查询（例如

462 [104] 中的 50%）将是独一无二的，因此不会出现在缓存中。也就是说，答案缓存的命中率有极限。这可以通过在搜索集群层缓存倒排索引列表来解决。这个缓存更有效，对于重复的查询项可以有超过 90%的命中率 [104]，即使查询是不同的。然而，因为结果缓存会快很多，所以对于每类缓存所要投入的比例并不容易计算，这依赖于系统。

早期的工作 [1329] 提出了使用建立在一组持续"最优的"历史查询上的查询日志，来提高未来相似查询的检索效果。后来，Markatos[1091] 证明了在查询中存在临时的局部性，并使用命中率作为指标，比较了最近最少使用（least recent used，LRU）策略不同变种的性能，结果发现静态缓存对于小的缓存是有用的。Cao[328] 提出考虑除了局部性之外的参数的缓存策略，如被缓存对象的大小和从磁盘获取对象所需要的时间。这个方法对索引进行如下组织：将预计算结果的索引（从过去的用户查询）和最频繁查询项的倒排表保存在主存内，其余部分的索引则保存在辅助存储器内。

由于系统通常是分层次的，因此提出了很多关于多级缓存架构的建议。Saraiva 等人 [428] 提出了一种使用二级动态缓存系统的 Web 搜索引擎的新架构。这项工作的主要目标是提高分层搜索引擎的响应时间。在他们的架构中，二级缓存都应用了 LRU 驱逐策略。他们发现第二级缓存能够有效地减少磁盘的流量，从而提高了整体的吞吐量。Baeza-Yates 和 Saint Jean[123] 提出了一个三级索引结构，带有基于项频的倒排表静态缓存。

根据 Markato 的观察，Lempel 和 Moran[1002] 提出了一种称为概率驱动缓存（Probabilistic Driven Caching，PDC）的新的缓存策略，它试图估计所有向搜索引擎提交的后续查询（从第二个结果页面开始）的概率分布。PDC 是第一个采用预取方法来应对用户请求的策略。为此，PDC 利用一个用户行为模型，其中用户的会话开始于请求第一个结果页面的查询，并能够随着一个或多个后续查询继续进行（即要求连续结果页面的查询）。当 τ 秒内没有收到后续查询，就认为会话结束了。

Fagni 等人 [544] 说明将静态和动态的缓存策略以及自适应的预取策略相结合，可以达到较高的命中率（称为启发式 SDC 算法）。在实验中，他们发现将很大部分条目投入到静态缓存并与预取结合会获得最高的命中率。另一方面，Zhang 等人 [1773] 研究了使用多种动态缓存算法来对压缩倒排表中的数据块进行缓存的方法，发现从内存中驱逐倒排表中最不常用的块，会在命中率方面表现得很好。

Baeza-Yates 等人 [104，105] 首次对静态和动态缓存的结果进行比较研究，特别关注于倒排表缓存和对结果的内存分配。他们发现，当把 30%缓存用于结果，而其余部分给倒排表时，可以获得最好的结果。他们还提出了一个用于静态缓存倒排表的新算法，它基于背包问题中的一个著名的启发式方法，用查询频率和倒排表长度的比率来确定应该缓存什么。图 11-8 显示了这个算法的结果，并与 LRU、LFU 以及之前的解法 [123] 相比较。它们还
463 展示了查询分布的变化是很小的，而且对可以定期（如每天）重新计算的静态解法影响很小。关于动态缓存的类似结果在 [1047] 中提出。

动态结果缓存的一个问题是，由以后不再出现的唯一查询所产生的污染效应。为了解决这个问题，Baeza-Yates 等人 [112] 提出了一种准入控制机制，它基于简单的预测来决定是否需要缓存答案集合。在这个解决方案中，缓存被分成两部分。第一部分用来缓存在未来可能会重复的查询结果；第二部分用来缓存其他所有的查询，以处理突发性查询和预测误差。无论是在第一部分还是第二部分对结果进行缓存都取决于查询的特征，如之前的出现频度或长度（按词或字符统计）。它们的结果在 SDC 算法的基础上有所提升，并说明了通过简单的特征，就可以减少缓存污染。

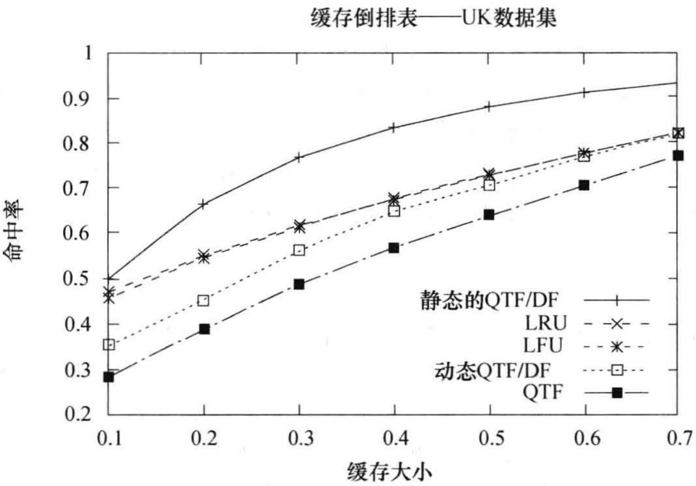


图 11-8 不同倒排表缓存算法的性能 [105]

11.4.4 多级索引

分层的索引代表另一种的改进。例如，考虑两级或两层的索引。第一级是较小的快速索引，用于最频繁的查询；而第二个级是较大的慢速索引，用于其他查询。在 [1359，1360，123] 中探讨了如何将文档集划分到多个层次。Risvik 建议为了可扩展性的目的采用多层结构 [1359，1360]。在这个系统中，层次充当过滤器，根据在给定层的命中数量以及该层查询结果的相关性得分，将查询"下传"到下一层。 464

上述技术的缺点是有些查询会有较慢的答案，尤其是当按照层次依次搜索时。一个解决方案是在所有层并行提交查询。然而，这增加了查询负载，因此总的硬件设施开销会增加。一个较好的解决方案是预测哪些查询需要传到下一层。为此，Baeza-Yates 等人 [114] 提出了一种基于机器学习的预测器，完全基于查询的特征来决定搜索是否应该并行地进行。他们说明比随机方法强的任何预测都能节省回答时间，但在硬件开销可能有负面的影响。因此，他们提供了关于性能-花费权衡的分析，表明每当回答时间有所减少时，都会在硬件花费上有小的增加。

Liu 等人 [1043] 对于多级索引进行了另一项研究，他们的实验表明，在中国互联网上抽取的语料上，可以减少语料规模 95%来生成"干净的语料"，但仍保持检索性能不变。更具体地说，90%的查询可以通过干净语料中的文档来回答。他们利用 PageRank（见 11.5.2 节）和入链数等查询独立的特征，将每一个页面分成潜在的检索目标页面（干净的语料）或普通的页面（被删除）。

Ntoulas 等人在 [1214] 中提出了另一个层次化技术，在缓存结果后使用剪枝索引来为简单的查询提供快速的结果。他们介绍了一个二层的架构，第一层是一个小的剪枝索引，第二层是搜索引擎的完整索引。查询首先通过小的剪枝索引回答。如果得到的答案和完整索引相比，在效果上没有损失，那么就将结果返回给用户；否则，就在完整索引上执行查询。在第一步中，这个方法似乎是有好处的，然而进一步的研究发现当考虑缓存的影响时，这个方法的价值就不那么明显了，因为剪枝索引和倒排索引缓存有着基本相同的效果，适应性却不如它们 [1487]。实际上，这些结果显示研究组件间的整体交互是多么的重要，而不应该孤立地改进每个部分。例如，结果缓存使实际命中搜索引擎的查询的平均长度增加了 30%，

同时显著减少了单个词查询。

对于如何在层次间划分 Web 文档集，D'Souze 等人 [514] 使用 TREC 语料第一、二卷中来自美联社的数据，比较了三种划分语料库的方法 [703]。他们尝试了随机划分、基于文档时间顺序划分，以及基于文档作者的划分，模拟一个可管理的环境，其中文档集是基于某些相同的特点划分的。同样，Craswell 等人 [437] 使用 wt2g 文档集，这是 wt10g 文档集的子集。该文档集从 956 个服务器上爬取数据，他们的数据比 TREC 的其他模拟数据更真实。

Yom-Tov 等人在 TREC Terabyte 评测任务中测试了 Web. gov 文档集的各种划分方案。将每个查询提交给所有的分区，最终的结果列表是通过对单个结果列表进行加权合并得到的，其中每个结果列表的权重基于估计每个分区回答查询的好坏程度的预测算法而获得。所测试的划分方案分别基于文档聚类、域、入链数和文档标题字符串。后者整体表现最好，然
465 而随机划分排在第二。它们应用查询预测算法来合并结果，而不是为层次选择做一个前期处理。另一个缺陷是，它们没有事先决定要搜索的语料库，所以在并行过程中每一层都会被检索。

11.4.5 分布式架构

爬取器-索引器架构存在多个变种，这里，我们将描述一些最重要的。它们中最重要的早期例子是 Harvest 获取架构 [244]。在最新方法中，我们主要介绍 Baeza-Yates[106] 等人提出的多站点架构。

1. Harvest 架构

Harvest 架构应用一个分布式的架构来收集和分发数据，这比标准的网络爬取架构（见第 12 章）更有效。其主要缺点是 Harvest 架构需要协调多个 Web 服务器。有趣的是，Harvest 分布式方法不受爬取器-索引器架构中某些常见问题的影响：

- 不同爬虫的并发请求所引起的服务器负载增加。
- 爬虫获取整个对象，造成了 Web 流量的增加，但大多数内容最终却没有保留下来。
- 信息由各个爬虫独立收集，造成引擎间缺乏协调。

为了避免这些问题，这里介绍架构中的两个主要组件：收集器和代理。收集器从一个或多个 Web 服务器中收集和抽取索引信息。收集的时间由系统定义，是周期性的（也就是系统的名字所表示的收获时间）。代理提供了索引机制和所收集数据的查询接口。代理从一个或多个收集器或其他代理检索信息，增量地更新它们的索引。根据收集器和代理配置的不同，服务器负载和网络流量会有不同程度的改进。例如，收集器可以在 Web 服务器内运行，而不产生外部流量。此外，为了避免重复的工作，收集器可以将信息发送给多个代理。代理也可以过滤信息并将信息发送给其他代理。这个设计允许工作和信息通过一个灵活和通用的方式来分享。Harvest 架构的例子在图 11-9 中显示。

Harvest 架构的一个目标是建立特定主题的代理，关注索引内容并避免通用索引中的词汇表和可扩展性问题。Harvest 架构包括一个专门的代理，允许其他的代理注册关于收集器和代理的信息。在建立一个新的系统时，这对于确定一个合适的代理或收集器是最有用的。Harvest 架构也提供了复制器和对象缓存。复制器能够用来复制服务器，提高基于用户的可
466 扩展性。例如注册代理能够被不同地理区域复制以便更快地访问。复制还可以用于在很多 Web 服务器间划分收集进程。最后，对象缓存降低了网络和服务器的负载，也减少了访问网页时的响应时延。更多细节可以在 [244] 中找到。

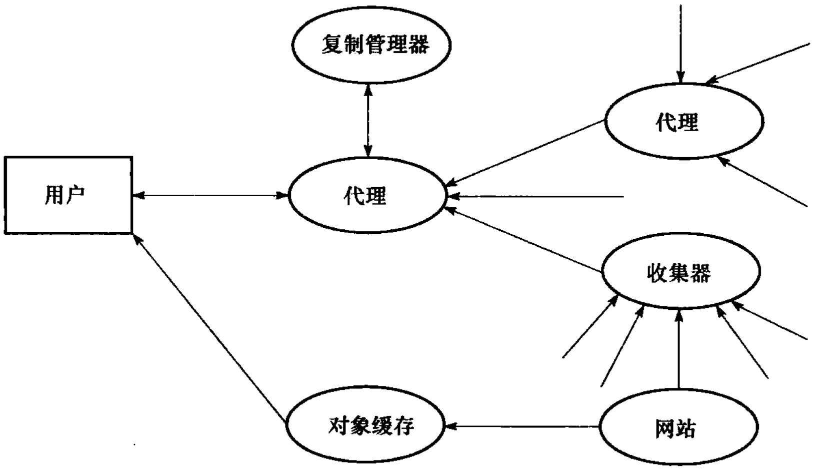


图 11-9 Harvest 架构

指向网页还是指向词语位置是索引颗粒度的一个指示。如果它指向逻辑块而不是页面，索引会不那么密集。这个方法可以通过使所有的块具有基本相同大小来减少不同文档的大小差异。它不仅减少了指针的大小（因为块数比文档数少），还减少了指针的数量（由于词语之间引用的局部性）。事实上，非频繁词语往往会共同出现在同一块上。这个想法在 Glimpse[1078] 中用到，它是 harvest 架构的核心 [244]。查询像在倒排索引中一样处理，获得块的列表，然后顺序搜索这些块。需要注意的是，精确的顺序搜索可以在 RAM 中以每秒 30MB 的速度完成。Glimpse 原本只用了 256 块，用来在 200MB 文本中对那些不频繁出现的词进行有效的搜索，且索引仅占文档的 2%。通过调整块的数量和大小，对于较大的文档集，可以得到合理的空间-时间的权衡（详见第 9 章）。这些想法还不能应用在 Web 上，因为网络访问，顺序搜索没办法负担得起。然而，在分布式体系结构中，如果索引也是分布式的，那么逻辑块就有意义。

2. **多站点架构**

随着文档集的增长，查询处理器的容量也需要增长，以匹配高查询吞吐量和低时延的需求。然而，单处理器性能的增长不可能和 Web 等超大规模文档集的增长相匹配，即使采用大量的服务器。最主要的原因是物理的和管理的限制，如单个数据中心的大小以及能源和制冷需求 [151]。因此，使用不同查询处理器的分布式查询解决方案是一种可行的方法，因为 467
它更具有可扩展性。然而，它同时也面对新的挑战。一个挑战是将查询分发到合适的查询处理器上，以便更有效地利用可用的资源，并提供更精确的结果。影响查询分发的因素包括地理邻近性、查询的主题或查询的语言。地理邻近性的目标是通过利用与提交查询的用户相接近的资源来减少检索时延。这种功能的一种可能的实现方法是 DNS 重定向：根据客户端的 IP 地址，DNS 服务器将查询分发到适当的 Web 服务器，通常是网络中距离最近的 [1539]。作为另一个例子，DNS 服务器可以应用地理位置来决定将查询分发到哪里。由于在一天内，一个特定的地理区域内提交的查询会有些波动 [168]，因此也可能从繁忙区域中的服务器卸下一些负载，将这些查询重新分发到一些不那么繁忙的区域。

考虑到上述的讨论，Baeza-Yates 等人 [106] 最近提出了一个搜索引擎的成本模型和一个简单的分布式架构，其成本与集中式搜索架构具有可比性。该架构基于星形拓扑结构逻辑相连的若干站点，这样中心站点是本地查询负载最大的站点。其主要的思想是在本地回答本

地查询，仅当在答案中需要外部页面时，才转发到其他站点。为了增加本地查询的比例，作者建议使用结果缓存，并复制一份所有站点内热门文档的小集合。这两个技术可以将本地结果的数量从 5%增加至 30%或者以上。

在一篇补充性论文中，Cambazoglu 等人 [325] 表明通过在本地回答查询，所节约的资源能够用作执行更复杂的排序函数，这样就可以改善结果。

11.5 搜索引擎排序

排序是搜索引擎必须执行的最难、也是最重要的功能。第一个挑战是制定适当的评价过程，可以从用户相关性方面来有效地评价排序的效能。没有这样的评估过程，就不可能很好地微调排序函数，产生高质量的结果。第 4 章已经详细介绍了很多可能的评价技术和指标。在 11.5.6 节，我们将在 Web 的背景中介绍这个话题，特别注重对于用户点击的利用。

第二个重要的挑战是 Web 内容质量的识别。质量可以根据多个信号来指示，如域名（例如，**.edu** 是一个积极的信号，因为来自学术机构的内容更有可能被检索），文本内容和各种计数（如词语共现数）、链接（如 PageRank）和被搜索引擎监管的网页访问模式。事实上，如之前提到的，点击是质量的关键因素。搜索引擎的流量越多，可用的信号也就越多。网
468 页的布局也提供一些额外的可用信号，如标题、元数据和字体大小等，在后面将进行讨论。

当前搜索引擎所采用的基于广告的商业模式在经济上刺激了第三个挑战的产生：避免 Web 垃圾。Web 环境中的垃圾制造者指的是企图人为地增加之前所提到的信号来欺骗搜索引擎的恶意用户。欺骗是可以实现的，例如将网页中的一个词重复很多次；使用链接农场；通过可见的着色技巧向用户隐藏索引项，但是对搜索引擎可见；最先进的方式甚至是欺骗性 JavaScript 代码。更多的细节在 11.5.7 节给出。

最后，第四个问题在于确定排序函数并计算排序（这与上面提到的质量评价不同）。虽然比较不同的搜索引擎是相当困难的，因为它们在不同的 Web 语料库上开发和运行，但是领先的搜索引擎必须使用它们自己的评价方式，不断地评价和自我比较，以保持竞争力。

在接下来的小节中，我们将讨论一些经典的信号（signal），在排序函数中它们通常被搜索引擎用做相关性的指示，或者在基于机器学习的排序函数学习中用做特征。之后我们讨论基于链接的排序函数，相比经典的信息检索，这是一个为 Web 而发明出来的领域。然后，我们从简到难讨论了三种不同的排序技术。最后，我们介绍了质量评价和 Web 垃圾。

11.5.1 排序信号

我们区分应用于改善排序的不同类型的信号，包括来源、内容、结构或用途。

*内容信号*和文本本身以及文档中词的分布相关，这是信息检索领域的传统研究问题。这种情况下的信号包括简单的词计数和 BM25 等完整的 IR 得分等（见 3.5.1 节）。它们也可以由布局提供（即 HTML 源），从简单格式指示（标题给予更多的权重）到复杂的指示，如页面中某标签的邻近信息。

*结构信号*是 Web 链接结构的内在信息。它们中的一些在本质上是文本，如锚文本，它以非常简短的形式描述目标网页的内容。事实上，锚文本通常用做所链接网页的代表文本。这意味着即使还没有爬取到网页，也可以通过搜索与它们相链接的锚文本来得到网页。其他信号涉及链接本身，如页面的入链数或出链数。我们应该注意到，基于链接的信号比经典的搜索引擎排序具有更广泛的用处。一个例子是应用基于链接的信号，在社交网络中回答名称查询，如 [1638] 所讨论的。由于这些基于链接的信号对于 Web 是典型的，但却没有用在

传统的信息检索系统中，11.5.2 节会充分介绍它们。

下一组信号来自 Web 用途（Web usage）。主要的一个是用户通过点击产生的隐式反馈。在我们的情况中，对点击最主要的利用是结果集合的 URL。这个重要的相关性资源已经在 4.5.5 节和 5.4 节中介绍。另外的信号包括用户的地理环境信息（IP 地址、语言等），技术环境（操作系统、浏览器等）和时序环境（通过用户 cookies 得到的查询历史）。这些特征可以给用户更好的本地化结果，并对其按地区或语言定制，甚至对它们进行部分的个性化。469 请注意，这个应用遵循幂律法则。也就是说，有少量重量级的用户和许多较轻量级的用户。因此，尽管个性化只对某些用户行之有效，但如 7.2.3 节所描述的那样，对许多用户共有的意图进行个性化更为简单和有效。

11.5.2 基于链接的排序

鉴于任何一个给定的查询能得到几千个甚至几百万个可用的页面，对这些页面通过排序来产生一个短列表可能是 Web 信息检索中最关键的问题，它需要一些相关性的估计。在这种情况下，指向页面的超链接的个数是其受欢迎程度和质量的度量。此外，网页之间可能有很多共同链接，不同的网页也可以被相同的网页所引用，这些都经常指示了网页之间的联系，对排序有潜在价值。下面，我们介绍几个利用链接的排序技术的例子，但在是否依赖于查询上有所不同。

1. 早期的算法

科学论文重要性的一个信号是文章的被引用数量，这是图书馆科学的研究人员研究很久的一个话题 [880]。基于这个想法，多位作者提出通过入链对网页进行排序 [1086，855，1021]。然而，人们很快就意识到只统计链接的个数不是很可靠的权威性测度（对科学论文引用也是这样），因为很容易通过建立新的页面来从外部影响这个数量（成本基本为零）。

Yuwono 和 Lee[1760] 在经典的 TF-IDF 方法（见第 3 章）之外，提出了三个排序算法，它们是：布尔传播（Boolean spread）、向量传播（vector spread）、最多引用（most-cited）。前两个是通常的基于布尔模型和向量空间模型的排序算法，并将它们进行扩展，考虑某个答案页面所指向的页面，和指向某个答案页面的页面。第三个算法，“最多引用”仅仅基于有链接指向答案页面的页面所包含的索引项。这些技术对比研究考虑了在 2400 个网页上的 56 个查询。结果显示向量传播模型得到更好的召回率-精度曲线，有 75%的平均精度。

另外一个早期的例子是 WebQuery[338]，它也实现了网页的可视化浏览。WebQuery 处理网页的集合（例如搜索结果的列表），然后基于每个网页的相连程度将它们排序。此外，它通过发现与原始集合高度相连的网页来扩展这个集合。一个相关的方法是由 Li[1021] 提出的。

2. HITS

一个更好的想法由 Kleinberg[911] 提出，应用在超文本推导主题搜索算法（Hypertext Induced Topic Search，HITS）中，这个排序方法是查询依赖的，它考虑指向答案页面或者被答案页面所指向的页面的集合 S。在 S 中，被很多链接指向的页面称为*权威页*（authorities），因为它们倾向于包含权威和相关的内容。有很多向外链接的页面称为枢纽页（hub），470 它们倾向于指向相关的相似内容。存在一个积极的双向反馈：较好的权威页面被好的枢纽页面所指向，较好的枢纽页面指向好的权威页面。令 $H(p)$ 和 $A(p)$ 是页面 p 的权威，值和枢纽值。定义这些值，使全部页面 p 都满足以下公式：

$$H(p) = \sum_{u \in S | p \to u} A(u), \quad A(p) = \sum_{v \in S | v \to p} H(v) \qquad (11\text{-}5)$$

表 11-2 不同国家的基于链接的指标中幂指数的总结

国家	PageRank	HITS	
		Hubs	Auth.
巴西	1.83	2.9	1.83
智利	1.85	2.7	1.85
希腊	1.83	2.6	1.83
韩国	1.83	3.7	1.83
西班牙	1.96	n/a	n/a

其中 $H(p)$ 和 $A(p)$ 对于所有页面是归一化的（在原始论文中，每个指标的平方和设置为 1）。这些值可以通过迭代算法得到，它们收敛于链接矩阵 S 的主特征向量。在 Web 情况下，为了避免 S 规模的爆炸，定义了指向答案的最大页面数量。这个技术在链接不存在、重复，或者自动产生的情况下不能很好地工作。一个解决方法是基于周围内容来衡量每个链接的权重。第二个问题是主题扩散，因为链接权重带来一个后果，即结果集可能包括与查询不直接相关的页面（即使它们有很高的枢纽和权威值）。这个现象的一个典型例子就是当一个特定查询被扩展到一个包括原始答案的、更广泛的主题。这个问题的一个解决方案是将每个页面的内容关联到一个得分，就像在传统的信息检索排序一样，然后将得分和链接的权重结合起来。链接的权重和页面的得分可以包含在之前的公式里，通过与和式的每一项相乘 [352, 199, 351]。实验结果表明，前 10 个答案的召回率和精度显著地增加 [199]。通过将链接划分成子组，并在子组上而不是原始网页上应用 HITS 算法，也可以利用网页中链接出现的顺序。表 11-2 中，我们展示了全球不同国家的权威值和枢纽值分布的幂指数，来自 [97]。

3. PageRank

最有名的基于链接的权重是 PageRank，它是谷歌 [253] 最初应用的排序算法的一部分。如下所示，PageRank 模拟用户在 Web 上的随机导航行为。考虑到用户正处于页面 a。
471 接下来，她通过随机选择页面 a 中的一个超链接，移动到一个页面 a 指向的页面。下一步，她在移动后的网页上不断重复这个过程。在多次的移动后，我们能够计算用户访问每个网页的概率。这个概率满足图的性质，在 Web 环境中称为 PageRank。真正的 Web 图中包含死端（dead end），即那些没有出链和自链接的页面。为了避免用户陷入这样的网页中，需要考虑另外一种情况，使得她也能以一个小概率 q 跳到图的任意一个其他页面。

因此，用户可以以概率 q 跳转到 Web 图中的一个随机页面，或者以 $1-q$ 的概率跟随当前页面中的一个超链接跳转。通过 Web 图上随机游走的定义，用户从来不会跟随已经经历过的超链接回到刚刚访问过的页面。这个过程可以建模成一个马尔科夫链，由此可以计算出在每一个页面的平稳概率。定义 $L(p)$ 是页面 p 出链的个数，假设页面 a 被页面 $P_1 \sim p_n$ 指向。那么，在页面 a 找到用户的概率 $PR(a)$，也就是页面 a 的 PageRank，定义为：

$$PR(a) = \frac{q}{T} + (1-q)\sum_{i=1}^{n}\frac{PR(p_i)}{L(p_i)} \tag{11-6}$$

其中 T 是 Web 图中页面的总数量，q 是系统中必须设定的参数（典型值是 0.15）。注意其他页面的排序（权重）被页面的链接数归一化了。PageRank 能够通过迭代算法计算，并对应于 Web 的归一化链接矩阵（即马尔科夫链的转移矩阵）的主特征向量。

在计算 PageRank 的时候，存在一些技术问题。主要的一个问题是关于处理马尔科夫链中的死端或者“沉没结点”，即那些没有出链的网页。一个解决方法是对所有这些页面，q 都等于 1。更简单的方法是去掉它们（这也可以减少链接矩阵的大小），最后再使用它们父

结点的 PageRank 来计算它们的 PageRank。

在［91，92］中，定义了一系列通过链接传播页面重要性的排序算法。这些算法应用了随距离减小的衰减函数，于是一个直接链接就比在长路径中的链接更为可靠。作者研究了三种路径长度的衰减函数：线性的、指数的，以及双曲线衰减。指数衰减对应于 PageRank，其他函数是新的。在其他结果中，他们说明如何计算一个线性逼近来得到页面的顺序，这与使用固定的、较少的迭代次数的 PageRank 是基本相同的。另外，他们也介绍了为什么 q 取 0.15 是个好的值。在表 11-2 中，显示了在不同国家 PageRank 分布的幂指数。

接下来是最后的历史评价。第一个关于超文本相关性的通用方案对相关性插值定义了一个线性微分方程，形式是 $y=r+Dy$，其中 r 是初始的相关性估计，D 是带权重的连接矩阵［286］。通过对 r 和 D 做一个合适的选择，我们能够从这个公式产生 PageRank。 472

11.5.3 简单的排序函数

最简单的排序方案由一个全局排序函数构成，如 PageRank。在这种情况下，页面的质量与查询无关，查询只作为一个文档过滤器。也就是说，满足查询的所有网页都按照 PageRank 的顺序排序。

更复杂的排序方案由不同相关性信号的线性组合构成。例如，将 BM25（见第 3 章）等文本特征和 PageRank 等链接特征相结合。为了说明这一点，假设页面 p 满足查询 Q。那么，给定查询 Q，页面 p 的排序得分 $R(p, Q)$ 可以这样计算：

$$R(p,Q) = \alpha BM25(p,Q) + (1-\alpha)PR(p) \tag{11-7}$$

而且，如果 p 不满足 Q，那么 $R(p, Q)=0$。如果我们假设所有的函数都进行了归一化，且 $\alpha \in [0, 1]$，那么 $R(p, Q) \in [0, 1]$。需要说明的是，这个线性函数对于 α 是凸的。另外，式子的第一项依赖于查询，而第二项不依赖查询。如果 $\alpha=1$，我们得到了纯粹的文本排序，这是早期搜索引擎的典型情况。如果 $\alpha=0$，我们得到了纯粹的基于链接的排序，它是独立于查询的。因此，对于包含查询 q 的页面，其顺序是事先已知的。我们可以应用标注数据或者点击数据作为正确答案，通过实验来调整 α 的值。实际上，α 可能是依赖查询的。例如，对于一个导航型查询的 α 值要比信息型查询更小（见 7.2.1 节）。

Silva 等人［1478］报告了一项较早的工作，将基于文本和基于链接的排序相结合。作者应用一个贝叶斯网络来结合不同的信号，说明了这种结合会比孤立地使用任何一种排序函数带来更好的结果。Calado 等人［311］在接下来的研究中讨论了基于链接的全局排序和局部排序在计算 Web 结果时的对比效果。对于网页 p，基于链接的局部排序方法只考虑了链接 p 的页面和被 p 链接的页面。作者将基于文本的排序方法（向量模型）分别与全局 HITS 算法、局部 HITS 算法，以及全局 PageRank 算法相结合，并对结果进行了比较。实验表明，基于链接的全局排序在排名靠前的部分得到更好算法的结果，基于链接的局部排序在排名较后的部分则表现得更好。

11.5.4 排序学习

一个非常独特的计算 Web 排序的方法是应用机器学习技术来进行排序学习。为此，人们可以使用他们喜欢的机器学习算法，依靠包含排序信息的训练语料，来“学习”结果的排序，这类似于用监督算法进行文本分类（见第 8 章）。在这种情况下，需要最小化的损失函数是学习算法中的错误数，类似于传统分类算法中错误分类的个数（见第 8 章）。对学习到的排序进行评估必须在不同于训练语料的其他数据集上进行（也包含排序信息）。对于查询

Q，有三种排序信息可以用于训练：

473 • 点式（pointwise）：一个与 Q 相关的页面集合。
• 对式（pairwise）：指示两个页面间排序关系的相关页面对的集合。也就是说，对 $[p_1 > p_2]$ 意味着页面 p_1 比页面 p_2 更相关。
• 列式（listwise）：排好序的相关页面：$p_1 > p_2 > \cdots > p_m$。

在任何情况下，可以认为包含在排序信息内的任何页面比不包含在排序信息内的页面更相关，或者我们可以保持那些情况不定义。另外，排序信息也不需要一致（例如，在对式中）。

训练集可能来自所谓的"编辑判断"，既可以来自手动标注，也可以来自点击数据，后者更好。考虑到用户点击反映了用户的偏好，而且在很大程度上与评测人员的相关性评价一致（见第5章），人们可以考虑通过点击信息来产生训练数据。这样，我们就能够从基于点击的偏好中学习排序函数。也就是说，如果对于查询 Q，p_1 比 p_2 有更多的点击，那么 $[p_1 > p_2]$。

从点击数据中使用对式方式的排序学习方法使用支持向量机算法（见8.2.1节）来学习排序函数。这是由［1320］提出的。这种情况下，偏好关系可以转化成由带权重项向量表示的排序文档间的不等关系。然后这些不等关系转化为一个支持向量机优化问题，它为文档中的项计算最优的权重。这种方法建议将带有不同权重的不同搜索函数结合到一个单一的排序函数中。

点式算法通过对于单个文档进行回归或分类来解决排序问题，而对式方法将排序问题转化为文档对的分类问题。这两种方法的优点是它们可以应用现有的回归或分类结果。然而，排序有一些内在的特点不能被对式方法解决。

列式方法直接解决排序问题，它采用列式损失函数，或者直接优化平均精度等信息检索评价指标。然而，这种方法通常比较复杂。有些作者提出应用多变量函数，也叫做关系排序函数，来代替基于单文档的排序函数，来求解列式排序。

损失函数的第二种可能性是用不同的松弛优化函数来最大化平均精度（见4.3.2节），由Joachims[841］和Yue等人［1759］提出。更多的技术细节可以在Chakrabarti的优秀综述［350］中得到，包括这个问题的其他变种。

为了研究，微软发布了一个带有排序信息的数据集LETOR[1039]，它被研究人员用做比较的基准而得到广泛采用。

11.5.5 学习排序函数

另一个不同的方案是学习排序函数，而不是学习排序的顺序。它等价于为基本排序学习
474 可能的最好函数。其想法是使用遗传算法，其中，种群的成员是在给定的排序特征集合上的函数实例。在遗传算法的每一步，不同函数变异或交叉，使用基准或训练数据来评估函数的适合度。在多次迭代之后，挑选出最适应的函数。

这个方法的一个明显的优点，它们可以通过简单地观察函数来发现重要的特征，以及它们对最终排序的影响。这种想法似乎同时独立出现在Trotmann[1595］的文档排序任务和Lacerda等人［954］的广告排序任务中。这两种方法应用相似的函数集（标准的运算符和典型的简单函数，如对数函数或指数函数），但是他们组合函数的方法，以及被优化的"适合度"函数的特点是不同的。

因为这个技术非常新，所以在改进结果质量以及效率方面还需要进一步的研究。

11.5.6 质量评价

为了能够评价质量，Web 搜索引擎通常使用人工评价指出对给定的查询哪个结果是相关的，或者从用户点击估计真实情况，或者最终选择两者的结合。

1. **前 5、10、20 平均精度**

评价 Web 搜索结果质量的一个简单方法是将标准的精度-召回率指标应用到 Web 中（见 4.3.2 节）。为此，以下观察是很重要的：

- 在 Web 上，几乎不可能评价召回率，因为对于典型查询来说相关页面的个数是非常多的，甚至是不知道的。因此，标准的精度-召回率数值不能直接获得。
- 大多数 Web 用户只查看前 10 个结果，而用户查看前 20 以外的结果是相对不常见的。因此，没有必要评价排在 20 名以后的 Web 结果的质量，因为它不反映一般用户的行为。
- 由于 Web 查询比较短和模糊，因此对结果的人工评价应该是由多个不同的人对每个查询-结果对进行相关性评价。例如，如果某个查询-结果对有三个单独的评价，其中至少两个评价显示是相关的，那么我们可以认为结果和查询确实相关。

这些观察的综合影响是：1）对于 Web 结果的精度，应该只评价排列靠前的位置，即 $P@5$、$P@10$ 和 $P@20$；2）每个查询-结果对应该有 3～5 个独立的评价。 475

2. **将点击数据作为评价指标**

应用点击数据来评价答案质量的一个主要优点是它的可扩展性。缺点是在较小语料库的情况下效果不太好，如 Web 内容很少的国家、企业网搜索，或者在查询的长尾区域（见 7.2 节）。注意，用户的点击并不是被当成二值信号来使用的，而是通过更复杂的方式应用，如考虑保持在所点击页面停留了较长时间（一个好的信号）或者从一个结果跳转到其他结果（这个信号表明没有找到满意的结果）。这些测度及其用途是复杂的，也是某些领先的搜索引擎的秘密。

为此，在这种情况下最主要的评价指标是基于大规模点击数据的前 10 或前 20 个结果的平均精度。利用点击评价的更多细节见 4.5.5 节。

使用点击时一个需要考虑的重要问题是，点击率被答案的排序（排名更好的网页得到更多的点击）和用户界面（例如，在第一页的最后一个结果和第二页第一个结果之间，点击数会有一个间断）所影响。因此，去除这种点击偏向来发现它们的真值是很重要的 [111，1321，522]。这个问题的更多细节在 5.4 节中讨论。另外，在正确的环境中获得点击也是很重要的 [1277]。

3. **评价片段的质量**

一个相关的问题是评价结果片段的质量。搜索片段（snippet）是从搜索引擎结果中产生的一小段文本摘录。它们提供了搜索结果的摘要，说明结果与查询的相关性（例如通过将查询项变成黑体）。它为用户提供了很大的价值，用户可以快速查看片段来决定哪个结果是他们感兴趣的。

既然搜索片段在显示搜索结果中扮演了重要的角色，那么对其质量的评价就很重要。实际上，近期关于这个领域的研究正在增加。在 [861] 中，作者研究了片段长度的变化是如何影响结果质量的。在 [873] 中，作者研究了如何预测搜索片段的可读性。在 [35] 中，作者提出了伴随时间信息的搜索片段，并评价了它是如何改进结果的。在所有这些研究中，最可取的评价技术是众包（crowdsourcing），特别是 Amazon Mechanical Turk（AMT）平

台，在4.5.4节已讨论了。

11.5.7 Web垃圾

Web蕴涵着很多盈利的商机，经济上的激励促使网站拥有者希望其网站在搜索引擎的结果列表中排在前列。所有希望提高搜索引擎排名的欺骗行为通常被称为Web垃圾（Web spam）或垃圾索引（spamdex）。与垃圾做斗争的相关研究领域称为敌对信息检索，它已经成为一些论文和研讨会的目标[23]。

Web搜索引擎一定要考虑到，“任何对网页的可重复特征计数的评价策略都是易于操纵
476 的”[1238]。实际上，这样的操纵是很广泛的，而且在很多情况下是成功的。[531]的作者报告，在1亿个网页的PageRank计算中，前20个URL中有11个是色情的，这些高排名似乎都是通过使用相同形式的链接操纵来得到的。

在[690]中，垃圾被定义为“任何为了得到与网页的真实价值相比不公正的相关性或重要性倾向的蓄意行为”。垃圾网页是用于直接进行垃圾行为的网页，或者得分因其他垃圾网页而人为提高的网页。[1256]给出了另一种垃圾的定义，即“任何欺骗搜索引擎相关性算法的企图”，或者，极端地说，“如果搜索引擎不存在，就不去做的任何行为”。

已有很多垃圾索引技术，随着垃圾制造者和搜索引擎公司之间的斗争，新的技术会继续被发明。一个垃圾网页可能包括一个出现次数异常高的关键词，或者包含其他一些通常与基于内容的垃圾检测技术[1216，512]斗争的文本特征。链接垃圾包括链接农场，它或者在同一个拥有者的网页之间创建一个复杂的链接结构，或者串通欺骗搜索引擎。点击垃圾通过特殊的软件机器人来实现，它发出特定的查询请求，然后点击预先选择的需要提升的网页。第三种更复杂的方法是程序垃圾，Web垃圾制造者在网页中注入JavaScript的代码片段[㊀]。当在客户端执行时，这段代码显示给用户的信息和从搜索引擎中爬取的不同（这是一种称为伪装（cloaking）的特定形式）。

有些人通常将Web垃圾和搜索引擎优化（Search Engine Optimization，SEO）相混淆。然而，当网络管理员按照大多数搜索引擎提供的指引，使他们的网页容易被发现时，这样的SEO技术是合法的。相反，恶意的SEO被Web垃圾制造者用于欺骗用户和搜索引擎。关于这个话题的另外一个绝佳的信息来源是Matt Cutt就SEO张贴在其博客上的内容[㊁]。注意，确定Web垃圾是Web挖掘应用的一个重要例子，相关话题在11.10.2中讨论。

11.6 管理Web数据

在本节中，我们主要探讨与搜索引擎需要存储和管理的Web数据相关的一些问题。

11.6.1 为文档分配标识符

文档标识符通常是随机设定的，或者依据爬取到的URL的顺序。数值标识符在某些数据结构中用来代表URL。除了倒排表外，它们可以用于Web图中的结点编号，以及在搜索
477 引擎资源库中标记文档。

在参考文献中已经表明，对文档小心地排序所形成的标识符分配方法，可以使索引和Web图存储方法都能受益[1333，222，209，1482，206，1480，1739]。同时，基于全局

㊀ Matt Cutts给出了关于欺骗性JavaScript的有趣例子，见http://www.mattcutts.com/blog/seo-mistakes-sneaky-javascript/。

㊁ http://www.mattcutts.com/blog/type/geogleseo/。

排序方案的标识符分配可以简化结果的排序（见 11.5.3 节）。

关于倒排表的压缩，如 Silvestri 在 [1480] 中所提出的那样，通过对所有引用文档集中 Web 文档的 URL 进行排序，可以得到一个十分高效的映射。对 URL 列表按照字典序的升序分配标识符能够提高压缩率。Silvestri 已经在经验上验证的一个假设是，共享某些关联或判别性词语的文档很可能被用在同一个站点中，所以它们的 URL 有较大的公共前缀。实验验证了这个假设：应用 URL 排序技术，压缩率可以提高到 0.4。而且，对数百万个 URL 进行排序只花费几十秒，并且只需要几百兆的主存。

11.6.2 元数据

正如在 11.4.1 节讨论的那样，200 亿个 URL 在使用压缩格式的情况下，需要至少 1TB 的空间来存储相应网页的元数据式。有效地管理如此大规模的信息必然需要快速和空间高效的数据库，从而又需要可用的高效文件系统。

谷歌的 BigTable[356] 可能是 Web 规模数据库最好的例子。BigTable 当前被用来在分布式系统中存储数据，通常应用 Map-Reduce 范式来生成和修改 [485]（见 10.5 节）。BigTable 不是一个传统的数据库，而是如作者所描述的“一个稀疏的分布式多维有序映射”，并且设计成可扩展到在“成百上千台计算机”上的 PB 规模。BigTable 可以在系统中加入机器，并且不需要任何重新配置就可以使用它们。

作为数据库，BigTable 兼具了面向行和面向列的数据库的特点。每个表有很多维，这些值以压缩的形式保存，它被优化成面向底层文件系统，如谷歌文件系统（GFS）[623]。

一个受 BigTable 启发的开源数据库是 HBase[725]。HBase 也是一个用 Java 编写的分布式数据库，它在 Hadoop 分布式文件系统（Hadoop Distributed File System，HDFS）[726，240] 上运行，为 Hadoop（Map-Reduce 的一个开源版本）[691] 提供了类似 BigTable 的功能。HBase 是面向列的，支持压缩，在内存中操作，并使用 Bloom filter。

其他可选方案包括 Hypertable[800] 和 Cassandra[341]。特别地，Cassandra 在一个类似于亚马逊 Dynamo 的基础设施上运行，因此能最终保持一致。Dynamo[486] 是亚马逊私有的键-值存储系统，它有很高的可用性，并且将数据库和分布式散列表的特性结合起来（见 10.8 节）。

11.6.3 压缩 Web 图

Web 图可以通过邻接表来表示。基本上，对于图中的每个结点 v，它们包含了从 v 可达的顶点列表。据观察，几乎所有链接的 80% 是本地的，也就是说，它们指向同一个站点的 478
网页。根据这个观察，为相同网站的 URL 分配相近的标识符（如 11.6.1 节讨论的），这将明显导致邻接表包含非常相近的标识符。对这些列表进行 d 间距压缩，会产生一个有很长的 1 间距的邻接表。

利用这一点和 Web 图特有的冗余可以达到极高的压缩率。压缩 Web 图的目的不仅是提供简洁实用的数据结构，而且为了更快速地访问，因为链接分析和其他应用需要 Web 图。举例来说，WebGraph 框架 [222] 将典型的 Web 图压缩到大概每个链接 3 位，在几百纳秒内为链接提供访问，并对这个主题的早期工作 [197，1333，14] 进行了改进。

11.6.4 处理重复数据

这个问题有两种。一种是检测指向相同网页，并且会在显示时加入冗余和噪声信息的多

个 URL（例如镜像）；另一种是检测指向部分重复内容的多个 URL（部分去重能够用来改善排序和检测垃圾）。识别重复也减少了需要检索和处理的文档集的大小。

对重复网页的定义并不那么显而易见。如具有相同正文但是不同 HTML 格式（或 CSS）的两个网页会有不同的布局。这样，如果我们要求重复页面的所有内容都相同，那么它们就不会被视为是重复的。事实上，大多数镜像系统的实现正是依据这个要求，它意味着可以通过对整个文档计算散列值来检测重复。所使用的散列函数应该很容易计算，冲突的概率也比较小（也就是说两个不同文档的散列值应该不同）。经常用于此目的的标准散列函数有 MD（Message Digest，消息摘要）以及 SHA（Secure Hash Algorithms，安全散列算法）。如果检测重复时不考虑格式，那么在去除 HTML 文档的所有格式指令后，相同的方法也可以使用。

近似重复问题处理起来更为复杂。近似重复的一个例子是仅仅因为修改日期或自动添加页脚而不同的镜像网页，通过散列无法检测出它们。检测近似重复的一个方法是使用余弦距离作为相似度度量（见 6.5.3 节）。Kolcz 等人［927］提出了一种优化方法，假设那些十分常见的词语会同时出现在每个文档中，因此需要被忽略。这相当于在统计意义上定义了一些禁用词。他们的报告显示，以返回相似度超过 90%的文档为目标，只使用那些在少于 5%的文档中出现的词，性能可以提高一个数量级。

另一个方法是使用 6.5.3 节中定义的类似度的方法。在这种情况下，我们能够选择函数 W（通常使用片段（shingle）），挑选出最佳阈值 t，以确保高效的计算［267］。在余弦和类似度两种情况中，如果两个文档的相似度（距离）大于（小于）阈值 t，就认为它们是重
479 复的。

有多个拟合距离的优化方法，它们在效率和错误率上有所不同，如 COPS[262]、KOALA[742] 和 DSC[267]。第一种方法是使用每一个片段的散列值，它基于 Karp 和 Rabin 用来搜索文本的文本指纹方法［877］。也就是说，散列值可以在线性时间上增量地计算。第二种方法是只考虑其中的某些片段，形成超级片段（super shingle）［742，269］。两种优化都减少了计算中需要比较的次数。事实上，如果每个文档的片段个数是不变的，比如 k 个超级片段，那么重复检测可以在线性时间内完成。

后来，Chowdhury 等人［380］提出了 I-Match 算法来计算每个文档的散列码，它不考虑太稀疏或太频繁的词。他们表明，在最坏的情况下，算法对于有 d 个文档的集合来说是 $O(d\log d)$ 级的，但在实践中是 $O(d)$ 级的。他们也表明它比超级片段算法表现得更好，部分原因是，如果小文档只包含常用的或者不频繁的词，那么它们是根本不用考虑的。在邮件垃圾处理［926］等应用中这是一个缺点，那里我们需要删除小的重复。另一方面，在 Web 中这可能不是个问题，因为小的重复文档不会经常出现在答案排序的前面。

11.7 搜索引擎用户交互

Web 搜索引擎服务于数亿个用户，他们中的大多数都很少具有技术背景。因此，界面设计已经被极简规则深深地影响。

> **极简规则**（Extreme Simplicity Rule）。用户搜索体验的设计，即用户与搜索引擎交互的模式，必须假设用户对搜索任务知之甚少，并且必须要求学习的东西越少越好。事实上，比起阅读搜索引擎的帮助页面，用户通常更愿意阅读新的冰箱或 DVD 播放器的“用户手册”。这种状况的一个直接后果是，当和搜索引擎交互时，没有“得到结果”的用户，很可能简单地转换到另一个搜索引擎来解决问题。在这

种情况下，极简性就成为 Web 搜索中用户交互的一个规则。

在本节中，针对最流行的搜索引擎，我们描述典型的用户交互模型、最近的创新，以及在这种极简规则下所面对的挑战。这里讨论的一些用户交互的概念已经在第 2 章中介绍过了，但是这里通过一些主要的 Web 搜索引擎，如 Ask.com、必应（Bing）、谷歌（Google）和雅虎（Yahoo!）等提供的用户体验更深入地研究。这里不讨论“垂直”搜索引擎，只限于特定知识领域的搜索引擎，如 Yelp㊀或 Netflix㊁，或者主要搜索引擎的垂直部分，如谷歌图像搜索或雅虎问答。 480

11.7.1 搜索矩形范式

用户现在已习惯于通过在搜索“矩形”中制定查询来阐述他们的信息需求。这种互动模式已经非常流行，以至于很多网站在主页的突出区域，都有一个矩形搜索框，即使搜索技术是第三方提供的。为了说明这点，图 11-10 显示了 Ask、必应、谷歌和雅虎搜索的搜索框。搜索矩形的设计一直非常稳定，谷歌的主页在过去的 10 年基本上都没有变化。Ask 和必应等其他引擎允许一些更花哨的设计，带有多彩皮肤和有趣的地点或对象的漂亮图像（例如图 11-10中 Ask 的金门大桥背景）。尽管有这些趋势，搜索矩形仍然是所有搜索引擎的中心。这个矩形通常也称为“搜索框”，如第 2 章所介绍的。



图 11-10　四个主要的搜索引擎将搜索矩形当做用户界面的中心组件（分别来自 Ask、必应、谷歌和雅虎。Ask screenshot，©IAC Search & Media，Inc. 2010. 版权所有。ASK.COM、ASK JEEVES、ASK 的 logo、ASK JEEVES 的 logo，以及其他出现在 Ask.com 和 Ask Jeeves 网站的商标归 IAC Search & Media 公司及其许可证颁发者所有）

虽然在搜索页面的中心显示搜索矩形是最受青睐的布局风格，但也有一些其他方案：

- 有些 Web 门户在主页的一个特别区域嵌入搜索矩形。这个方法的例子由 yahoo.com 或 aol.com 提供。
- 许多网站有一个高级搜索界面，它为用户提供一个由很多“矩形”和选项组成的表单（很少使用）。
- 搜索工具栏被大部分搜索引擎作为浏览器插件提供，或内嵌在 Firefox 等浏览器中，可以看成中心搜索矩形的精简版本。由于在任何时候都可以访问，因此它们是主页搜索矩形的一个更方便的替代品，但它们需要下载安装，这阻碍了更广泛的应用。注意，为了弥补这种开销，很多搜索引擎会与 PC 分发或制造商协商，提供优惠价格的 OEM，从而预装他们的工具栏。

㊀ http://www.yelp.com。

㊁ http://www.netflix.com。

481 • 由谷歌 Chrome 的“omnibox”所推出的“终极”矩形，合并了地址栏和搜索栏的功能。由浏览器来决定用户输入的文本是想要浏览某个网站还是进行搜索。在 omnibox 推出之前，Firefox 已经提供了一个功能，用来识别出某些词可能不是 URL 的一部分，应被视为查询的一部分。在这些情况下，它会触发谷歌的“手气不错（I feel lucky)”功能，返回搜索结果前面的部分。有趣的是，这个 Firefox 功能是可以定制的，允许用户触发谷歌以外的搜索引擎，或者获得完整的搜索结果页。

1. 查询语言

如 7.1.1 节解释的，用户通过一个词语序列来表示他们的查询。事实上，在“自由文本”查询和布尔查询的捍卫者很多年的论战之后，Web 搜索引擎基本上赢得了这场战斗。因此，自由文本格式已经变成 Web 搜索引擎事实上的标准查询语言。现在的用户通常输入一个词语序列来描述他们的信息需求 [268] 或目标 [1382]。有些搜索引擎宣称查询的隐含语义是所有词的“AND”操作，谷歌等其他引擎则宣称“每个词是有关系的”。大多数搜索引擎保留按需改变语义的权利，因此额外的操作符，如“+”不保证在未来是否会存在。

事实上，尽管所有搜索引擎的大多数主流查询语言是词语的序列，但 Web 搜索引擎都默认一个基本的查询语法。查询语言通常由“+”、“-”和“site:”等一元操作符来限定紧随的词，OR 等二元操作符来操作前导和后继的词，或者由双引号等分隔符来表明精确的短语匹配。

大多数搜索引擎没有公布正式的查询语言，而是在它们的网站上提供了搜索提示或选项。表 11-3 比较了一些领先的搜索引擎在关联页面中提供的提示/选项，其中包括 Ask.com [76]、必应 [202，203]、谷歌 [652] 和雅虎搜索 [1736]。这个表没办法非常详尽，因为有些查询操作符是在隐藏模式下实现的，而且在其他地方公布的。这样的例子有谷歌的“numrange（数值范围)”功能 [650]，它是在 2004 年宣布的，它允许用户通过在两个数字间输入“..”来指示数字的范围。例如，查询“DVD 播放器 100..300 美元”匹配 [100，300] 范围内的任何一个数字，例如 250。在表 11-3 中，我们关注于众所周知的“稳定”操作符，而不是我们刚才描述的、像“numrange”那样，具有强大的处理功能的操作符。

表 11-3 常见的查询操作符

操作符语法	细节	谷歌	雅虎搜索	必应	Ask
“..”双引号围绕一个字符串	短语搜索	是	是	是	是
+前面是一个空格，处理后面立即跟随的项/短语	这个操作符保证相关联的项在结果中按照现在的样子被包含	是	是	是	是
-前面是一个空格，处理立即跟随的项/短语，必应也使用 NOT	这个操作符保证相关联的项不出现在任何结果中	是	是	是	是
OR（|）处理出现在前面和后面的项或短语	和布尔 OR 等价	是	是	是	是
site：后面跟随站点名	从特定站点返回结果	是	是	是	是
hostname：后面跟随主机名	从特定主机返回结果	否	是	否	是
url：后面跟随 URL	检查下面的 URL 在搜索引擎中是否存在	否	是	是	否
inurl：后面跟随一个项	返回 URL 中包含特定项的结果	否	是	否	是
intitle：后面跟随一个项	返回标题中包含特定项的结果	否	是	是	是
inlink：/inanchor：后面跟随一个项	返回链接或锚文本元数据中包含特定项的结果	是	否	是	是

除了这些常见的操作符外，下面列出了一些由单一搜索引擎所支持的独特操作符。关注它们的用途，并确认随着时间的推移，它们是否被其他搜索引擎采用，这将是很有趣的。

- Ask 的时间操作符：
  - □ afterdate:，beforedate:
    上面的操作符后面跟随一个 yyyymmdd 格式的日期，分别返回在给定时间之后或之前发生的查询结果。
  - □ betweendate:
    这个操作符以类似的方式工作，但是接受由两个逗号隔开的 yyyymmdd 格式的日期。
  - □ last:
    这个操作符跟着一个给定的时间区间，有下面 6 种取值：{week，2weeks，month，6months，year，2years}，返回在这个特定时间区间内找到的结果。
- 必应：
  - □ And/&
    这个操作符只有必应搜索提供，它表示前导和后继词语之间的布尔“AND”。很有趣的是，必应提供了这个操作符，尽管它宣称“缺省的搜索都是 AND 搜索”。 482~483
  - □ ()
    括号是用来将词语和－或＋等其他操作符组成一组。这个功能增加了其他操作符的能力，但估计是为那些有数学或逻辑背景的用户所保留的。
  - □ 必应搜索有很多独特的操作符，例如 filetype:、contains:、ip:、feed:、prefer:等，有些看起来很有前途，有些则没有。
- 谷歌：*
  通配符代表一个缺失的完整词语（而不是词的一部分），表明对于搜索引擎，它应该被视为一个“对于任何未知词语的占位符”。
- 雅虎搜索：link:跟随着一个 URL
  此操作符返回链接到给定 url 的文档，这是由雅虎站点浏览工具所提供的一个功能。本质上，比起操作符，它更应该被认为是一种到达另一个功能的快捷方式。事实上，雅虎搜索作为一个内容提供商，在快速到达其功能方面，比大多数其他的搜索引擎更强。它允许通过一些保留标记直接访问，默认的保留标记可以通过在搜索框中输入!list 得到。它们包括如!news、!flickr、!wiki、!map。

从上面的例子可以看出，可能是根据使用的情况，主流的搜索引擎提供了更少的操作符。可以猜测，它们很少被使用，Web 事实上的标准查询语言就是自由文本。不过，这可能会改变，因为后来者会创新，并试图通过新的功能来赢得市场。同时，并不令人吃惊，在本书写作的同时，作为市场中最新的搜索引擎，必应搜索提供了最广泛的操作符，可能期待着用户通过使用来表明他们的喜好。

2. 动态查询建议

动态查询建议服务，在第 2 章也称为“自动填充”或“自动建议”，通过交互功能丰富了搜索框。当用户在搜索框中输入字符时，查询建议通过下拉列表将建议提供给他们。谷歌和雅虎提供的例子在图 11-11 中显示。

这种服务是谷歌建议（Google Suggest）开创的，由 Kevin Gibbs 发明，2004 年部署在谷歌实验室，不久后出现在其工具栏中，但没有出现在谷歌的主页上，这可能是由于扩展性

的问题。2007 年，雅虎搜索走在谷歌的前面，在 yahoo. com 和 search. yahoo. com 上部署了搜索助手功能。有趣的是，雅虎放弃了之前经典的前缀完成模型，因为它也提供了中间字符的填充，雅虎搜索为前缀“inform”所提供的“webmd health information”搜索建议如图 11-11所示。谷歌建议随后也在谷歌主页上推出，首先在 youtube. com，然后 2008 年 8 月也出现在 google. com 上。2009 年年初，谷歌建议最终部署到了所有的谷歌域名上。其他的
484 搜索引擎在快速地追赶，类似的服务也出现在美国的 Ask 和必应搜索上，甚至在 Amazon. com 等具有足够流量的垂直搜索服务中。注意，如第 2 章讨论的，Netflix 或 nextbio 等垂直搜索引擎的查询填充服务是语料库驱动的，而不是查询驱动的，因此不能扩展到 Web 上。

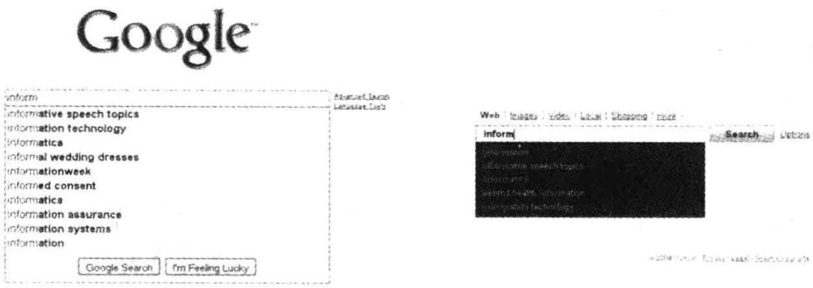


图 11-11 谷歌建议和雅虎搜索助手

今天，大多数用户享受了这些服务的便利，而不关注它们。按照我们前面提到的极简规则来说，这对于搜索引擎用户交互范畴来说可能是至高无上的赞美。

动态查询建议应该区别于只在查询发出后运行的查询建议系统。动态建议系统需要很少量的信息，它们的主要输入是前缀而不是一个良构的查询。不过它们在处理过程中可以应用搜索引擎可以使用的所有额外信号，只要这些信号既可以利用又不影响响应时间。这些额外的信号包括可用的用户历史和个性化特征、地理信号，也可能是同一用户会话中之前的查询。

动态查询系统不是全新的，类似的功能在 Emacs[1527] 等早期编辑器也曾有提供，它支持命令填充或 Kom shell 等 shell 脚本 [224]，根据要求，当用户输入一个制表符或空格字符时，会填充命令以及文件/目录名。之后在移动领域，也提供了这样的辅助服务来减少在烦琐键盘上需要输入字符的数量 [66，1233]。这些早期的特征与现代的动态查询建议最主要的不同是：

1）现代建议语料库的来源和规模。

2）性能要求，因为现代的建议服务需要同时服务大量的用户（而不是在单个应用/设备上的单个用户）。

3）用户体验的智慧，根据用户输入自动地触发现代建议，而不是通过请求。

随着 Web 环境中搜索流量和性能的不断提高，这两个主要变化使应对这些挑战成为可能。事实上，搜索流量令人难以置信的增长使得搜索引擎聚集了巨大的查询语料，可以用来为所有的用户服务。通过使用查询日志而不是手头的语料库作为主要的数据源，可以使用更
485 接近的语言模型来给用户展示更“自然”的建议，这就增加了精度。此外，在 Web 搜索中巨大的性能提高允许集中化的服务，其平均负载（qps，即每秒处理查询数）5 倍于常规搜索。在默认情况下，每当输入一个新的字符时，就会发送一个请求给服务器，相关的建议需要很快返回，因为用户仍然正在输入查询。

有趣的是，要给用户带来价值，动态建议服务必须克服经典信息检索的两个挑战。1）效率：尽快地返回建议，使得建议可用；2）有效性：给出最相关的建议。

- **效率**。快速的响应时间通常是通过两种途径实现的。首先，如 Ji 等人［834］所提出的，与给定前缀相关联的候选必须以一种有效的数据结构预先计算和存储，以保证在查询时最少地处理。其次，它们需要快速服务，通常使用分布式数据中心来完成。最后，搜索引擎中嵌入的 JavaScript 建议代码应该尽可能地精简，并且为安全起见，不要阻止用户立即给出查询，以防止花费比预期更多的时间。
- **有效性**。给定在矩形框中输入的字符串，从极端情况下的一个到几十个字符，动态建议服务基本返回 5（如搜索主页上的雅虎助手）到 10 个候选查询（如谷歌建议或雅虎结果页面）组成的列表。相关性的主要标准是流行度，不仅反映在查询日志中的出现频率，还根据（可能的话）点击率等附加信息来反映。需要考虑很多技术问题来确保相关性，如下所示：

□ **自动拼写校正**。给出一个错误拼写的查询建议是十分奇怪的，即便它比较流行。事实上，因为查询日志语料库显著地小于 Web 语料库，所以，虽然“群体智慧”［1546］原则对于下面讨论的拼写功能有很好的作用，但对于查询日志的长尾部分可能就没有那么有用了。这个“小语料库挑战”在有相对较少 Web 内容的国家或者在企业搜索中比较常见。它不仅影响查询的自动拼写修正建议，而且还会影响某些通过群体智慧建立的其他功能，如本节介绍的那样。

□ **过滤不适当的建议**。不适当的查询，通常包括色情或仇恨，仍然在查询日志中占有很大的比例。它们需要被过滤，以避免伤害用户以及遵守当地的法律。此外，比起在常规搜索的情况，垃圾是一个更尖锐的问题，特别在查询的长尾部分，因为它们不经常被观察到，更容易欺骗。

□ **查询建议去重**。当提及一个十分流行的主题时，可能只有细微的不同，但表达相同信息需求的查询，虽然在相关性方面都是符合的，但是没有给用户增加任何价值，因此是恼人的。如复数形式和词语的不同顺序。动态建议努力识别这些准重复 486
(quasi-duplication)，并从建议框中识别最有代表性的部分。

□ **建议的多样性**。这个问题是与前一个主题相关的，但是一个更微妙的方式。对于一个给定的前缀，查询填充的某些含义可能占据了主导地位，以至于罕见的含义可能没有机会得以显示。这就会有一个危险：流行的查询变得更流行（富者更富的情形），因此，对于有趣的主题，可能会看到其相关性得分缓慢地下降，直到它们因为缺少点击而最终从建议列表中去除。

□ **新鲜度**。热门的主题是有趋势的，通常跟随新闻出现，用户不能等待动态系统重新生成自己的索引，以使这些热门主题可见。因此，需要区别对待新鲜的建议，如索引动态搜索语料库时的情况。

□ **个性化**。这是指用户和区域的个性化。用户可以在浏览器的搜索框中看到他们过去的查询历史，他们可能不愿意丢失它们。然而，如果用户没有登录，同时显示个人查询和社区查询也许是不可能的，甚至给人留下隐私被侵犯的印象。社区/区域个性化更为关键，因为兴趣和流行度在不同的位置和文化间相差很多。对于前缀“real”，第一个查询建议是“Real Madrid”，而不是“real player”，在美国的用户可能会惊讶，然而在西班牙的用户如果没有得到这样的结果才会震惊。在最低的要求下，根据国家的个性化建议是非常重要的。这也可能需要越来越精细地对地理位置进行个

性化。事实上，当在阿拉斯加的用户给出查询“pizza”时，如果仅仅因为其流行度不同，第一个查询建议是“pizza palo Alto”或者“pizza Manhattan Upper West Side”，那么他可能会非常恼火。技术挑战是，当在越来越小的语料库上工作时，需要搜集足够的证据来保持高精度。

总体而言，动态建议的有效性能够通过覆盖度和质量（召回率和精度的一种）来衡量。取得很好的覆盖度，也就是说用户总是希望将建议框中分配的空白填满，当前缀变长时，这就变得极其棘手。前缀越长，命中长尾查询的可能性越高，这样就没有充分的证据来将查询选为建议。质量显然也是必须的，因为用户期望查询服务可以“读懂他们的心”。大多数用户会错误地认为，查询建议是“搜索引擎的声音”，而不是实际中“用户社区的声音”。被名字推荐所冒犯的一些个人和公司提起的诉讼 [1009，1096] 和负面新闻说明了这一点。

因此，我们认为动态建议服务会保留下来，并且是一个值得进一步研究的、有前途的领域。它们在搜索引擎页面占据了极其突出的地位。与十年多以前的最早版本相比，谷歌建议
487 实际上是谷歌主页上最容易看到的变化。它们代表了一个明显的瓶颈，因为提供错误的建议会给很大部分[⊖]用户带来“坏的查询”，这可能导致固有或广告搜索的结果变差，因此会影响整个收入流。

此外，它们的成功实现需要克服许多经典信息检索的挑战。例如，读者应该考虑上述问题的有效性以验证如下事实：如果用“查询”，取代“前缀”，以及用“结果”取代“查询”，它们将使读者想起经典的检索问题。在查询建议的情况下，这些挑战以一种更加极端的形式体现，并带有更严厉的有效性和效率的约束。

11.7.2 搜索引擎结果页面

1. 结果表示

(1) *基本布局*

如第 2 章中讨论的那样，搜索引擎结果页面 (Search Engine Result Page，SERP) 的经典显示风格包括一个“固有的”或“算法的”结果，出现在结果页面的左侧[⊜]，以及出现在右侧的支付/广告结果（广告）。此外，最相关的支付结果可能会出现在首页北部区域中固有结果的顶部，如图 11-12所示。尽管必应等一些搜索引擎允许用户自定义在一页中显示的结果数量，但默认情况下，大多数搜索引擎在首页中显示 10 个结果。

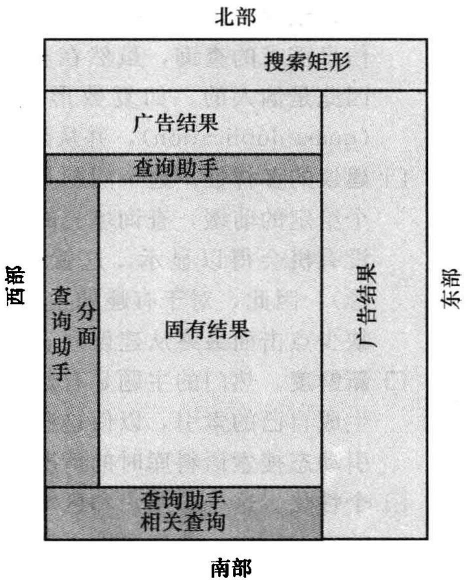


图 11-12 一个典型的 SERP 布局

图 11-12 展示了大多数搜索引擎所采用的布局，尽管有些例外，但它们大都有相同的、用虚线框表示的位置。这些搜索引擎可能在某些小细节上略有不同，如“查询助手”功能可能出现在页面的北部、南部，或西部区域；导航工具可能显示在西部区域，但也可能不显示；拼写校对建议的位置可能在北部

⊖ 大多数搜索引擎不公布其推荐服务的点击率，但是有理由相信是两位数。

⊜ 对于从右向左书写的语言，页面右侧是固有结果，左侧是广告结果。

区域广告结果的前面或者后面。

搜索引擎不断地试验布局的一些小变化，而未来可能采用截然不同的布局，因为这是一个需要创新功能的地方。例如，Cuil[⊖]介绍了一种完全不同于一维排序的布局，但是这更多地是一个例外，而不是应该遵循的规则。相反，主搜索引擎之外的搜索功能常常是不同的，以谷歌和雅虎的图像搜索，或者谷歌的广告搜索结果［655］为例，它们都跨越多个栏来显示结果。

本节仅专注于搜索结果中的固有部分。从现在开始，我们将把它们称为“搜索结果”，请注意与支付/广告搜索结果的区别。 488

（2） 标题/片段/URL 实体

主流搜索引擎使用非常类似的格式显示单独的结果，它们基本上包括：1）用蓝色或下划线显示的标题；2）由从结果页面抽取的两三个句子组成的简短片段；3）指向包含页面全文的 URL。在大多数情况下，标题可以直接从页面中抽取。当一个页面没有标题时，可以用指向它的锚文本生成一个标题。

如第 2 章中讨论的，片段由自动生成的摘要组成，其目的是在页面中强调与用户查询相关的主题。片段有助于决定是否需要点击一个链接。片段背后关键的问题是，它们需要在运行时产生，因为它们是查询依赖的。另一个非常重要的问题是，查询词通常在片段中通过粗 489 体突出显示，这需要一个字符匹配操作，或者当没有明显选择的时候，需要智能地找到合适的词。例如，考虑 Ben Gomes 在谷歌博客中给出的“arod”例子［637］，其中“在搜索结果片段中 Alex 和 Rodrigues 是用粗体显示的，它基于这样的分析：你很有可能指的是他”。Amitay 等人［49］介绍了一个用于生成页面的 Web 摘要的原创方法，它依靠指向它的锚文本。

当多个结果来自同一个站点或域时，搜索引擎通过缩排同一站点中较不相关的结果，将结果组合起来。最近比较出色的方法是“站内链接”（Sitelink）或“快速链接”（Quicklink）模式，其中“在搜索结果页面，导航快捷键［..］被显示在网站主页的下面，并让用户直接跳转到网站内的选定点”［347］。自动确定这些链接是不简单的，因为其目标是最大化大多数用户的利益，即在有限的区域显示相关的链接。然而，领先的搜索引擎似乎做得很好，并且可能会继续提高，因为最好的链接通常会通过点击和工具栏数据学习用户行为来推断出来。

（3） 更多的结构化结果

除了主要的 Web 语料库提供的结果外，搜索引擎还包括其他的结果类型。

- **“直接”（onebox）结果**。这是对应于可能有唯一结果的精确查询的非常具体的结果。由于其高度相关性，它们以独特的格式显示在常规的 Web 结果的上面。onebox 结果被用户查询中指示某个明确意图的查询项所触发。它们的目的或者是直接显示出结果，或者是显示出结果的链接，这可以提供终极的搜索体验，但是只有在非常特殊的情况下才能完成，即当相关性被保证、且答案简短而不含糊的时候。

  例如，谷歌和雅虎搜索都支持天气的 onebox 查询，通过输入“天气〈地点〉”来触发。（见图 11-13 中的例子）。搜索引擎支持的 onebox 搜索的一些例子包括：体育直接搜索、包裹跟踪（如 UPS 或联邦快递）、计算器、电影列表以及上映时间（尝试“Movies Palo Alto”）和火车时刻表等。一个更加耐人寻味、更有挑战的例子

⊖ http://www.cuil.com。

是“事实提取”，虽然它不按照经典的 onebox 搜索格式显示，但它也可以提供相似的体验。例如，在谷歌上查询“谁是加州的州长”，第一个结果会显示“加州-州长：阿诺德·施瓦辛格”。另一种类似于 onebox 的结果类型能够被没有唯一结果的查询触发，它是查询的一种模式。在谷歌上查询“从纽约到伦敦的航班”，会显示一个输入框，让用户输入确切的出发和返回日期，然后将新的查询转发到 Kayak 和 Expedia 等聚合器。谷歌在搜索功能页面列出它目前大多数的 onebox 搜索功能，以及“define:”等一些保留符号［654］。直接搜索基本上是一个快速解决（quick hack），
490 并且在某种意义上是常规排序列表方法无法满足所有用户需求的证据。

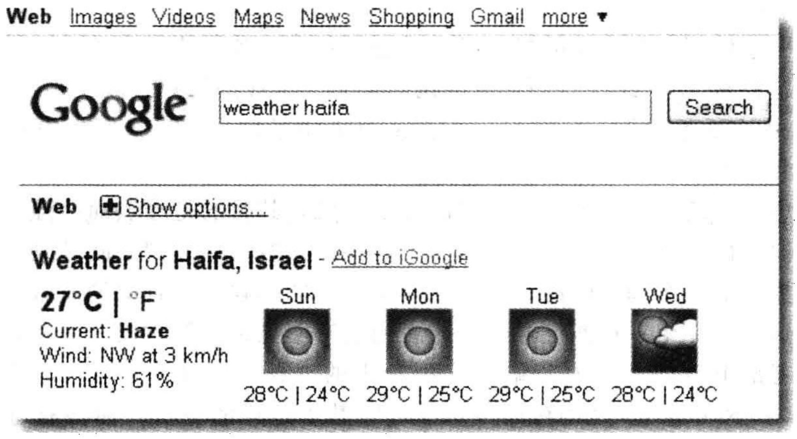


图 11-13 谷歌的天气 onebox 查询，不需要点击就能得到完整的答案

一个相关的概念是雅虎快捷搜索，它是部分手动干预的、对于热门查询（尝试城市或名人）的 onebox，位于固有结果的顶部。Marissa Mayer 在谷歌博客［1102］中讨论了一个更优雅的解决方案：“统一搜索”，我们接下来讨论它。

- **统一搜索结果**：除了提供核心 Web 搜索外，大部分 Web 搜索引擎还提供了其他的功能，如图像、视频、产品和地图，它们来自于自己的垂直搜索。虽然用户可以直接进入这些功能对特定语料库进行搜索，但“统一的”愿景指的是用户不需要指定目标语料库。搜索引擎应该猜测他们的意图并从最相关的恰当数据源中自动返回结果。主要的技术挑战是选择这些来源，并决定从每个源中显示多少结果。由于这是一个经典的联合搜索问题，因此使用 Fagin 和 Wimmers 提出的组合正交结果技术［543］是有帮助的。如今，搜索引擎不公布它们的方法，所以对于统一排序是否使用原则性的公式，或者是否使用一些启发式的方法都不是很清楚。通常情况下，如“小甜甜布兰妮”（Britney Spears）这样的查询会返回图像、视频和新闻，以及主要的 Web 结果。

Web 搜索结果可以以不同的格式出现，这取决于它们的类型。一个例子是 Ask.com，它直接显示了来自于雅虎问答（Yahoo! Answers）或维基问答（WikiAnswers）等问答系统的答案，而不显示片段。更普遍地，一个常见的趋势是对于来自主要 Web 语料库或者其他服务等特定来源的结果，以稍微不同的格式显示。Searchmonkey 说明了这一点，它是雅虎搜索的一个开放平台，允许发布者“和雅虎搜索共享结构化数据，以便显示一个标准的增强的结果”［1735］。SearchMonkey 在 2007 年被推出，它部分受到了 Peter Mika 关于微格式（microformats）的早期研究的启发。例如，雅虎搜索上的所有维基百科结构以 Search-
491 Monkey 格式显示。谷歌最近跟随了雅虎的范式，在 2009 年推出富搜索片段时，也探索了

富结果格式 [629]。最后，必应投入了很多的精力在结构化结果上，提供了特定领域的“文摘型”结果。下面列出了一些必应的例子。

- 旅行搜索结果带有票价趋势功能。
- 购物搜索结果包括一些便利的快捷方式，指向用户评价、专家评价、产品细节和价格比较、结果中最好价格、等级、易用性、可承受性、视觉线索和独特的“必应现金返还”。
- 健康搜索结果指示权威来源，如作为内容供应者的 Mayo 医学中心。
- 对于“总体”、“大气”等主题，本地搜索结果包括带有视觉信息的基于评价的分数卡。

很有趣的是，我们可以观察在保留相同类型的丰富功能的前提下，必应搜索是否容易地泛化、扩展到其他垂直领域及其他国家。

2. SERP 上的查询帮助

一旦用户提出查询，并查看搜索结果页面，他们的信息、导航以及事务型需求 [268]（见 7.2.3 节的更多细节）可以被：

- **满足**。这可能立即发生，比如当用户从计算器、天气、体育比赛结果等 onebox 搜索结果直接得到答案；或者几乎在它们点击排在前列的一个或者多个结果后立即发生。
- **部分满足**。这通常发生在用户进行“研究型任务”[218]，但没有一个单一网页拥有所有需要的信息时。11.7.2 节描述的搜索记事本等搜索工具，被很好地设计用来收集、注释和整理这些部分答案，以便形成一个连贯的单元，然后存储起来供日后使用，或出版/分享给别人。有些需求更容易触发研究型任务。这些例子包括用户寻找酒店、饭店的旅行需求，以及娱乐设施、功课、学生写作业的教育需求，或者有关疾病、症状和治疗方案的患者健康信息。
- **完全不满足**。这可能是因为用户没有很好地制定他们的查询，或者由于相关性内容不存在。对于搜索引擎来说，仍然不可能判断什么时候索引中不存在相关性内容，默认情况下，大多数搜索引擎会假设第一种场景，通过查询助手帮助用户重新制定他们的查询。

本节深入探讨了如何使用查询助手工具帮助用户精化或重构制定查询，如第 2 章所讨论的。 492

（1）拼写帮助

查询帮助最成功的例子是谷歌现在提供的著名的“您是不是要找”（Did you mean），它不同于通常的基于字典的模型，它彻底改变了拼写校对。事实上，经典的方法是使用编辑距离来识别字母倒置等打字错误 [944]（见 6.5.3 节）。而“您是不是要找”则简单地从大量使用中学习拼写校对。它广泛应用查询日志分析来进行拼写校对 [637]。谷歌发言人在校园演讲中给出的一个常见的例子是“Britney Spears”，查询日志显示她的名字有数百种错误的拼写方式（用户很有创造力），但是迄今为止最频繁的拼写是正确的那个。也就是说，群体智慧是最好的（见 11.10.2 节）。这种纯粹的频度信号对于长尾查询或者在日志不是很大的领域是不那么有效的，受到了前面提到的“小语料库挑战”的影响。在这种情况下，可以使用需要较少证据的其他信号。例如，前谷歌首席信息官 Douglas Merrill 在他的“101 个 Google 搜索技巧”中 [1121]，解释了搜索引擎可以通过简单地观察用户在两个连续查询中重写的查询来学习查询的正确拼写。Cucerzan 和 Brill 在 [461] 中研究了这个方法，展示了如何从查询日志中的查询重写来学习查询校正模型。

（2）查询建议

SERP 上查询帮助的其他手段还有查询建议。SERP 查询建议不同于搜索框中提供的动

态查询建议，因为它们能够利用更丰富的信息，包括良构的完整查询（而不是部分的）以及丰富的结果集合，带有各自的片段和相关的信息。查询建议通常在某种意义上与原始查询相关，当用户不知道如何表达他们的需求时最有用，他们转向相关的、并且可望制定得更好、结果也更好的查询。

已经有了一些关于挖掘查询日志来生成查询建议的研究工作。它们大致可以分为三类。

- **内容感知（content-aware）方法**依赖于搜索结果或者目标页面。这类工作早期的例子是 Raghavan 和 Sever [1329] 所做的，他们试图通过结果集文档的不同顺序来衡量查询相似度。虽然这个方法的优点是文档集能提供更丰富的信息，但它也带来了可扩展性方面的挑战。类似地，Fitzpatrick 和 Dent [567] 用前 200 个结果的归一化交集来衡量查询相似度。他们的技术同样受到可扩展性的影响，因为使用不同同义词的语义相似查询的交集通常是很小的。Sahami 在 [1406] 中使用了基于结果片段的查询相似度。他们把每一个片段作为查询提交给搜索引擎，希望找到包含原始
493 片段中查询项的文档。然后，他们使用这些返回的文档，为原始片段创建一个上下文向量。
- **内容无关（content ignorant）的方法**由 Befferman 和 Berger [166] 很好地描述，他们从被经常点击的 URL 推断出查询间的相似度。遗憾地，这个方法的影响力在某种程度上是有限的，因为结果页面中的点击数是相对较小的 [89]，因此相关的查询间距离矩阵是很稀疏的。通过使用大量的查询日志（如果这是合法的话），可能会使这种稀疏降低。
- **查询流方法**考虑到用户的序列搜索行为，以便更好地理解查询的意图。Fonseca 等人 [571] 和 Zhang 等人 [1777] 是这种方法比较好的例子。Fonseca 等人将查询日志看成事务的集合，其中每个事务代表一个会话，即单个用户在一个给定时间区间内提交的一个相关查询序列。这个方法有较好的结果，但也产生了两个问题。首先，它很难确定由属于相同搜索过程的连续查询组成的会话，其次，无法发现由不同用户提交的、最有趣的相关查询。

大多数现代的解决方案使用混合方法以获得更高的精度。例如，Baeza-Yates 等人 [109，110，111] 使用被点击的网页的内容为每个查询定义一个查询项权重向量模型。他们考虑在查询后点击的 URL 中的项。每一项通过查询的出现次数和出现查询项的文档被点击的个数来给定权重。一个看上去相当有希望的研究方法是从查询流中挖掘关系。

例如，会话通常是物理会话，而不是逻辑会话。所以，在很短时间内的四个连续查询可能是与两个完全不同的任务相关。通过最近在制定查询流图的尝试 [218]，我们期望看到更多的、能够得到更好结果的先进挖掘技术。

Web 搜索引擎不交流它们青睐的方法，但我们可以认为，它们使用了“最佳”的那个方法，并利用了多种信号。值得注意的是，对于在 SERP 上放置这些建议的位置上并没有达成一致意见。这是一个有趣的现象，因为位置对于使用有着直接的影响，并可能是衡量一个搜索引擎信任其建议工具的指示器。

谷歌在 SERP 底部的“相关搜索:”（Search related to:）标签下面显示查询建议（图 11-12底部的矩形）并以 4 列 2 行的形式安排它们。因此，可以预计这个功能的点击率是相对比较小的。有趣的是，谷歌“百宝箱”（tool belt）最新推出的“搜索选项”（Search options）功能提供了对于“相关搜索”，以及对原创（还没有广泛应用，可能是由于它精度比较低）的“神奇罗盘”（wonder wheel）的访问，给出了一个相关搜索项的图形化表示。

通过点击罗盘上的任何结点，用户在交互的动画罗盘上获得相关的主题，而结果在右侧保持更新。

雅虎搜索也在多个位置显示相关结果，如搜索框的右下方，在标签“您是不是要找”(Also try) 下面（见图 11-12 顶部的第二个矩形），在左侧的导航窗格中，甚至在搜索矩形内与常规的动态查询建议并排地显示。后者和常规的查询建议在不同的区域内，仅当用户继续输入查询或者自愿地扩展它时，才出现在 SERP 的搜索矩形内。最后，必应搜索在导航窗格的西部区域显示它们，并将它们标记成“相关搜索”(related searches)。 494

(3) 通过分面的查询精化

根据分面搜索范式，查询也可以通过将结果限定在一定的“分面”上来进行精化。分面搜索是一个导航机制，“通过文本搜索并逐步缩小在每个维度的选择，以使用户能够导航到一个多维信息空间”[279]。我们在这里将分面导航考虑成一个查询精化机制，因为在实际中它需要通过用户来选择分面。这个用户提供的输入使查询增加了额外的信息，从而能够更好地指定用户的需求，缩小了结果集。在第 2 章中，我们详细地回顾了分面导航，说明了分面导航在搜索系统中的应用，如 Flamenco、芝加哥大学的 Aquabrower 和 Yelp. com 等垂直搜索服务。

在 Web 搜索中，分面导航只是刚刚出现，由于规模的原因，有很多技术性挑战。在 Web 背景中实现分面导航的一种方法是，将结果的属性，如它们的类型（视频、音频）或者它们的来源（维基百科、YouTube、雅虎问答）映射到导航分面。这个方法被雅虎搜索采用，如图 11-14 所示，西部的导航窗格显示了一些用来缩小查询“巴塞罗那”(Barcelona) 的相关来源。必应搜索也使用类似的方法，如图 11-15 所示。虽然这种机制的实现细节还没有公开，但人们能够设想的一种简单的实现方法是，索引存储这些静态的属性，而搜索引擎在运行时获取与处理它们。



图 11-14 雅虎搜索中来源驱动的高层分面

谷歌通过之前提到的百宝箱，提供了相似的功能。用户可以通过各种类型的分面将结果“切片和切块”，分面从类型/来源，如“视频、论坛、评论”(Video, Forums, Reviews) 分面，到基于时间的分面，如“过去的 24 小时”(Past 24 hours)，“过去的一周”(Past week) 和“过去的一年”(Past year)。

一个更复杂的情况包括显示每一个分面的结果数量。还没有搜索引擎提供这种功能。虽然它在过去已经被如 Endeca [534] 等企业分面搜索引擎以及很多购物网站提供，但在 Web

495 的规模下，在一个适当的响应时间内估计这些数量是很困难的任务。

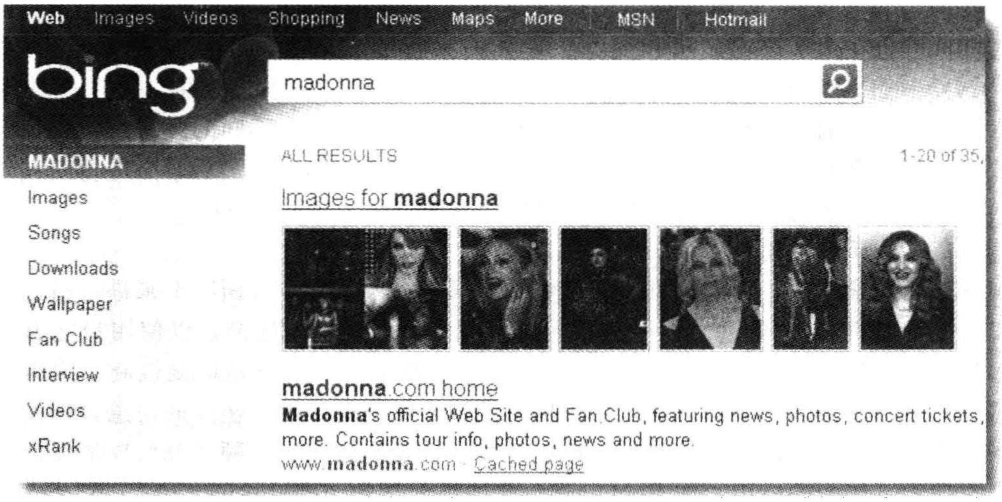


图 11-15 必应搜索左侧的分面导航

分面搜索研究已经调查了甚至更复杂的情况，如层次式的、关联的，甚至在运行时需要动态计算的分面 [1751]，以及与它们关联的可视化界面 [739]。然而，因为它们还没有被任何主流搜索引擎部署，所以在这里我们不讨论它们。跟随这个领域的进展是很有趣的，因为如上所述，主要的 Web 搜索引擎都在以各自的方式探索这个话题。

3. 可执行的结果

这里我们讨论的最后一个 SERP 话题涉及许多有前途的工具，比起简单的解释或者导航，它们允许对搜索结果进行更多的操作。这个列表无法面面俱到，因为这个领域很丰富，而且还在不断地发展。我们料想更多的工具很快出现，而且也有很多工具会消失，如 2009 年年初被谷歌停止的记事本扩展（notebook extension）[653]。

有些功能简单地对结果本身进行操作，用于各种目的，如雅虎搜索，或者谷歌“网页快照”(Cached) 链接和谷歌“类似结果”（Similar）链接。一个更高级的功能是谷歌“翻译此页”(Translate this page)，它显示在结果标题的旁边，将目标页面翻译成用户默认的语言。

另一种工具现在已成为 Web 搜索引擎的一部分，它是内置搜索框，通常显示在站内链接或快速链接的下面，这些链接允许用户在答案所在的网站内进行搜索。例如，在谷歌页面上对《New York Times》(纽约时报）的搜索会产生一个指向该报纸主页的链接，以及这样一个关联的站内搜索框。在框中输入的查询会以一个“site：nytimes. com”限定符来补充查询（见表 11-3 中“site:”的操作符），它将结果限制为只是这个网站内。

更有趣的工具包括谷歌搜索之星（Stars in Search）和雅虎搜索记事本（Search Pad)。“Stars in Search”功能在 2010 年 3 月推出，替代了更复杂但是可能不太成功的搜索维基
496 (Searchwiki)，搜索维基允许用户对于任何结果通过结果旁边的 3 个小图标提供反馈：气泡用于“评论”，箭头用于“推广”，“X”用于“删除”。利用这个现在已经停止的功能，用户可以按他们的意愿注释、推广和去掉结果。如果用户在未来重新提出相同的查询，他们会看到结果的个性化得到了保留。此外，用户可以通过点击 SERP 底部的链接，以便在任何时间访问他们自己的搜索维基笔记。搜索维基最近被精简版本搜索之星取代，它保留着搜索维基“推广”功能的一个变种。也就是说，三个小图标被一个单一的星星取代了，当用户选择了它，它就变成黄色，作为喜欢结果的一种标记。因此，对于后续的相似搜索，用户将看到之前加了

星标的答案出现在结果列表顶部的特殊区域中。这里需要记住的一个有趣的教训是，所有搜索引擎需要仔细监控使用情况，并修改或者完全去除那些没有得到足够吸引力的功能。

雅虎搜索记事本是一个有趣的功能，因为它和之前提到的谷歌记事本属于同一类，只是使用了不同的方法。搜索记事本允许用户轻松地跟踪他们咨询过的结果，并对它们整理和注释，以便后续使用或与别人分享。这种概念不是新的，它首先由 Bharat [196] 提出。但是使它变得独特、并可能比其他相关工具更有用的原因是，只有当搜索引擎确定用户是调查一个主题，而不是寻找快速的“一次性”的结果时，才会触发搜索记事本工具。所访问的页面能够被自动增加到适合的搜索记事本工具，而不需要用户像早期研究工作那样特别地“标记”它们，例如 Ask 的“My Stuff” [75]，以及已经不再维护的谷歌记事本 [939]。

11.7.3 培养用户

我们已经讨论过界面如何通过越来越丰富的片段以及结果切片逐渐进步，从而能协助用户进行查询描述和结果解释。然而，我们应该期望用户，尤其是年轻的一代，变得越来越有因特网意识，并能够对搜索引擎过程进行更多的控制。

按照指南，高级搜索界面向用户提供更好的控制查询效果的形式。通过在查询内部使用高级操作符，可以得到几乎相同的效果。为了更多地控制，高级用户可以指定尽可能多的查询项，并指示哪些查询项应该包括在结果中（通过“+”操作符）而哪些不应该（通过“−”操作符）。然而，用户不应该增加一个词所有可能的同义词，因为很少有网页对于一个给定的概念会使用多于一个或者两个同义词。如果用户能够把搜索限制在一个区域（例如，页面标题），限制某些属性（日期、国家）或者用表 11-3 中提到的操作符，那么结果集的大小必然会减小。

即使我们能给出好的查询，但结果集仍然可能非常大。考虑到前面提到的可视化工具对一般大众还是不可用的，并且它们是否会被一直采用还不是很清楚，所以用户必须从经验中学习。有很多策略来快速地发现相关的答案。如果用户正在寻找一个机构，他们总是能够尝试猜测相关的 URL，首先通过 **www** 前缀，后面跟随一个所猜测的机构的首字母缩写词或简称，最后跟一个顶级域名（国家代码，或者对美国来说的 **gov**、**com**、**edu**、**org**）。如果这还不能正常工作，那么用户可以在 Web 目录中搜索该机构的名称。 497

另一种有些常见的工作是用户在一个特定的主题上搜索出版的作品。为了完成这一任务，一个可能的策略是：

1）选择与主题相关的文章，如果可能的话，选择不常见的作者姓氏或者标题关键词。

2）使用搜索引擎来找到所有包含那些姓氏和关键词的网页。很多结果很可能是相关的，因为它们指向如下内容：a）引用那份原始文献的较新的论文；b）作者的个人网页；c）关于这个主题的页面，它指向很多相关的参考文献。通过在搜索过程中将最初的文章修改成更好的参考文献，这个策略能够进行迭代运行。

Web 提出了太多具有挑战性的问题，以至于有时更有效的方式是培养用户如何恰当地从搜索引擎和 Web 目录中获益，而不是试图猜测用户真正要什么。鉴于搜索引擎对 Web 的覆盖率有所不同，有一个方法是使用多个引擎或者元搜索引擎。这里的关键教训是：1）伴随着“针”，搜索引擎仍然返回太多的“干草”；2）Web 目录没有足够的深度来找到这些“针”。因此，当查询时，我们建议使用如下的经验规则尝试：

- 专门的查询：查看百科全书，这是它们存在的原因，不要忘记图书馆。
- 广泛的查询：使用 Web 目录来找到好的起点。
- 模糊的或者探索性的查询，以及反复的精化：使用 Web 搜索引擎，基于相关答案来

改进查询。

11.8 浏览

本节介绍另外一种发现范式——浏览，特别关注于 Web 目录。在大部分情况下浏览是有用的，例如当用户不知道如何指定查询（在全球互联网的环境下，这变得越来越罕见），或者当他们想探索一个特定的集合，但不确定其范围时。如今，浏览不再是 Web 上首选的发现范式。尽管如此，在特定的环境下它仍然是有用的，如在企业网或垂直领域。

在浏览的情况下，用户愿意投入一些时间来探索文档空间，寻找感兴趣的，甚至出乎意料的参考资料。无论是浏览还是搜索，用户的目标都是发现信息。然而，在搜索中，用户的目标更清晰。相反，在浏览的时候，用户的需求通常更广泛。虽然这种区别不是在所有情况下都存在，但为了简单起见，我们在这里采用了它。我们首先描述了 3 种浏览类型，即扁平(flat)、结构驱动（特别关注 Web 目录）和超文本驱动。然后，我们尝试将搜索和浏览相混
498 合的方式。

11.8.1 扁平浏览

在扁平浏览中，用户探索一个以扁平形式组织的文档空间。例如，文档可能由二维平面的点来代表，或者由一维列表中的元素来代表，这些元素通过字母或其他顺序排列。然后用户将在各处扫视，在所访问的文档中寻找信息。值得注意的是，对搜索结果的探索也是一种扁平浏览的形式。每篇文档也可以通过在浏览器中使用导航箭头和滚动条，以扁平的方式进行探索。

缺点是在给定的页面或者屏幕中，对于用户正在浏览的文档，可能没有明确的上下文指示。例如，当浏览大型文档时，用户可能忘记了他们正在看文档的哪个部分。由于其规模和分布，扁平浏览在全球互联网显然是不可用的。但是当浏览较小的集合时，它仍然是一个可选择的机制。此外，它可以用来结合搜索来探索结果或者属性。

实际上，扁平浏览一般在初始搜索后进行，并允许确定新的感兴趣的关键词。这些关键词可以被添加到初始查询中，以尝试提供更好的语境。这个过程是第 5 章所详细讨论的相关反馈的一种变种形式。

11.8.2 结构导向的浏览和 Web 目录

一个更加可扩展的浏览模型是结构驱动模型，其中层次结构或树结构等潜在的结构被用于浏览这个空间。这个模型在 Web 的早期是很流行的，当时搜索引擎的效果不好。目前它仍然应用于全球 Web，如雅虎目录 [1737] 或者开放目录计划（Open Directory Project, ODP，也被称为 DMOZ）[1220]。目录是将属于关联主题的文档组合起来的类别层次结构。有些目录针对特定的垂直领域。例如，有些网站专注于商业，有些网站专注于研究文献（例如 CiteSeerX[387]）。根据其应用领域，Web 目录（Web directory）可能也称为目录（catelog）、黄页或者主题目录。

表 11-4 展示了一些 Web 目录使用的第一级目录，其中第一级类别的个数在 12～26。有些子目录在 Web 目录的主页上也是可用的，这样就增加了 70 多个主题。最大的目录，如之前提到的 ODP 和雅虎目录，覆盖了数百万个网站。在大多数情况下，页面必须提交到 Web 目录，然后它们被检查并分到层次结构的一个或多个类别中。ODP 被认为是第一个 Web 2.0目录，因为它采用了协同对等模型（collaborative peer model）的方法，其中人们自愿担任编辑者。值得注意的是，即使潜在的目录结构是分层次的，但它们并没有形成真正

的树，因为交叉引用是很频繁的。因此，在实践中目录是有向无环图。

Web 目录还允许用户在分类描述或者分类指向的网页上进行搜索。事实上，因为分类网页的数量非常少，所以考虑到效率因素，目录甚至可以为所有网页保持一个本地的副本。代价就是目录必须确保时效性。

表 11-4　Web 目录中第一级类别的例子

艺术和人文科学	商业与经济	计算机与互联网
教育	娱乐休闲	游戏
政府	健康与健身	家庭
投资	儿童和家庭	生活与风格
本地	新闻	人物
哲学与宗教	政治	参考资料
区域	科技	购物及服务
社会与文化	体育	旅行及旅游业

目录浏览的主要优点是它提供的信息通常是有价值的。但另一方面，它也有两个缺点，第一个是分类总是不够专业，问题很多，并且目录提供的 Web 覆盖度是很低的（覆盖了所有网页的不到 1%）。因此，在 Web 规模，精度通常是可以实现的，但是召回率却不能保证。大多数 Web 目录意识到这个召回率问题，也发送查询给全球搜索引擎（通过一个战略联盟）来补充它们的结果。 499

目录的另一个问题是内容的增长，由于 Web 的不断扩大，这个问题每天都变得更严重。通过聚类或者其他技术，自动生成目录的努力在十多年前就开始了。然而，这种努力是很昂贵的，由于现有自然语言处理技术的限制（还远远没有办法有效地抽取关键概念），没办法对于所有情况都真正地成功。因此，大部分分类仍然是通过有限数量的编辑手动完成，这减慢了目录的增长。手动分类还有另一种限制，即缺乏术语一致性，这不仅发生在用户和编辑之间，也发生在编辑之间。

相同的结构导向模型能够应用在单个文档上。例如，当浏览电子书时，内容的第一级是章，第二级是节，最后一级是文本本身（扁平）。一个好的用户界面可以以聚焦的方式在层次结构中上行或者下行，帮助用户追踪文档上下文。

浏览树型结构时的一个常见的问题是当用户在深入某个路径中时，可能会丢失上下文。追寻上下文的一个常见方法是*面包屑路径*（breadcrumbs）或*面包屑小径*（breadcrumbs trail）[215，1244]，它出现在所访问页面的顶部，显示用户到达页面的路径。在小径上的每个位置通常是可以点击的，允许用户通过一次点击回到之前的位置。在图 11-16 底部给出了一个面包屑路径的例子，其中的路径是“目录>娱乐>游戏>视频游戏”（Directory>Recreation>Games>Video Games），它代表 4 个（可以点击的）步骤，指引到当前的“聊天和论坛”（Chat and Forums）页面。

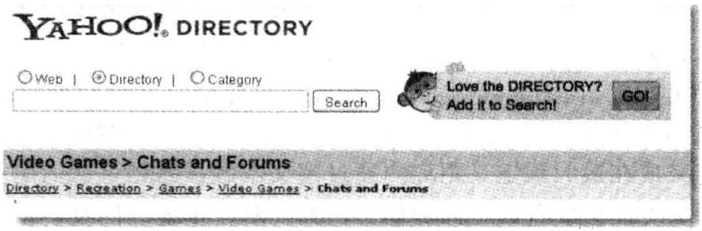


图 11-16　雅虎目录中面包屑路径的例子

一个相对成功的结构驱动模型体现在网站地图上，在大多数网站中，它或者显示在一个专门的网页中，或者在左侧导航窗格中。网站地图（Sitemap）早在 20 世纪 90 年代末期就已经开始研究了，关于 Site Mapping 的 WWW 研讨会说明了这点 [1068]，但是十年后才出现一些标准化。今天，大多数关键参与者都认可以 XML 形式表达的网站地图协议，如 sitemap.org 所公布的。然而，这些地图大多数用来帮助网站管理员与搜索引擎进行交流，

500 并告知他们哪个页面能够被爬取。

除了网站的真实结构，浏览可以被访问历史所指导。事实上，今天的大多数浏览器都提供历史地图。第 2 章和第 3 章中提供了浏览大型结构的更多细节。

11.9 浏览之外

11.9.1 超文本和 Web

由于历史的原因，许多人仍然认为 Web 是一个巨大的分布式超文本。这是一个误解，因为 Web 缺少内在的数据模型、导航规划和设计一致的用户界面。数百万的网页设计人员独立设计带有自己特色的界面。作为一个简单的例子，当在一个给定的网站上寻找电话号码时，我们经常找不到，因为它被埋葬在网站最意想不到的地方。因此，没有任何暗藏的力量帮助 Web 用户搜索感兴趣的信息。

因此，我们更倾向于将 Web 看成一组（部分）相互关联的网站的集合。其中的一些网站被认为是本地的超文本（在它们内在结构有某些一致性这个意义上），但是其他的可能仅仅是一些单独设计的网页集合（例如大学的网站内，各个院系设计自己的网页）。尽管缺乏结构和建模，但 Web 在交流上给我们提供了一个新的层面，因为它可以在一个非常低的成本下对全世界进行访问。更重要的是，Web 没有控制机构来设立监管和审查制度。因此，个人可以通过大型媒体发布其著作，而不用经过编辑委员会的过滤，对于人类历史来说这是第一次。也就是说，Web 是电子出版时代的最初标志，如第 1 章讨论的。

11.9.2 搜索与浏览相结合

将搜索与浏览范式组合的混合发现模型也已经开始研究，在不同的情况下有了或多或少的成功。如前面所提到的，大多数 Web 目录重定向一小部分查询到搜索引擎，或者在内部允许用户在内部搜索目录中的一个特定子树。相反，现在的搜索引擎将小的链接结构关联到
501 某些结果上，如 11.7 节所提到的片段上的快速链接。

如果 Web 结构，即它的超链接，在动态搜索方法中能变成搜索查询的一部分（如在 Web 早期的一些查询语言所支持的）或者搜索过程的关注点，那么搜索和浏览可以以一种极为不同的方式结合。这些超链接驱动的方法在研究上仍然受到限制，没有得到广泛的应用，主要由于多种局限性，如性能较差，可扩展性较差，或尴尬的用户体验。

过去，研究人员尝试了多种方法来结合这两种模式。WebGlimpse[1077] 是一个早期的例子，它在每个 HTML 页面的底部附加了一个小的搜索框，从而使搜索过程覆盖相邻的页面或整个网站，但仍然保留着浏览模式。这相当于跟随通过邻域搜索动态建立的超文本链接。网页的邻居定义成在最大预定义距离内通过超文本链接路径可达的网页集合。这个距离对于本地和远程网页可能会设置成不同的值。例如，对于本地的网页会很大，而对于远程网站，限制在一个距离值之内，如 3。邻居也包括网页所在目录的所有子目录。通过计算得到一个由网站或文档集的所有邻居组成的图，对于每个网页，有一个包含它所有邻居网页的文件。当搜索时，整个索引中的任何查询能够与邻居列表求交集，产生相关的结果。这种在一个给定种子周围动态爬取的方法来自 Mapuccino[754，1067]，又被 Fetuccino 扩展 [178]。另外，这些工具增加了可视化功能按需显示小的搜索/浏览地图。事实上，Mapuccino 允许用户依照自己的兴趣（以查询来表达）动态地产生网站地图。依照用户兴趣剪裁的导航地图在相关方向具有“更长的臂”，而彩色编码的结点指示了更多的相关内容。Fetuccino 扩展了

Mapuccino 的工作，主要通过更先进的基于 XML 的可视化方法以及两步搜索过程，该过程允许用户在区别领域查询和领域内的聚焦查询。这个工具发展为 IBM Websphere 的网站管理和站点分析工具。同样，很多网站地图工具演变成商业化的网站分析工具。这些例子包括，早期的 NetCarta 工具演变成微软的 SiteAnalyst、Dynamic Diagrams 的 MAPA、Surf-Serf、Merzcom 的 Merzscope、CLEARweb、Astra SiteManager、InContext 的 WebAnalyz-er、以及 SmartBrowser 的 HistoryTree。问题是对于搜索结果的可视化没有一个确定的标准，尽管早期提出了一些基于 XML 的方案，如 [34]。

还有其他一些使用可视化的浏览工具，它们可以被宽泛地分成两类：为可视化 Web 子集而设计的工具和为可视化大型答案而设计的工具。这两种情况都需要以合理的方式表示很大的图。可视化 Web 子集的非商业性的例子包括 WebMap[504]、Sitemap、Ptolomeaus，以及很多更早的研究 [52，530，1139，1161]。我们不讨论更通用的可视化软件，其中 Web 可视化只是一个特定情况，也不讨论其他有关的可视化工具，如 Web 用途分析（Web usage analysis）等 [619，1274，1513]。可视化大型答案的方法已经在第 2 章中介绍过了。

总之，尽管可视化的浏览工具通常非常赏心悦目，但它们还没有在整个 Web 中部署，因为它们还没有证明对用户提供的额外价值。 502

11.9.3 Web 查询语言

直到现在，我们都专注于以每个网页内容为目标的查询，而没有考虑直接查询连接网页的链接结构。例如，我们可能希望搜索至少包含一个图像，并且从给定网站通过至多 3 个链接可以到达的所有网页。为了允许这种查询，不同数据模型已被使用。最重要的模型，一个是表示网页（结点）和网页之间的超链接（边）的标记图模型，另一个是表示网页内容的半结构化的数据模型。在后一种模型中，数据模式并不总是事先知道，可能随着时间推移而变化，可能会很大并且仅是解释性的 [5，294]。

尽管在 Web 出现之前，已经出现了一些查询超文本的模型和语言 [167，416，1138]，但第一代 Web 查询语言是为了将内容和结构相结合（见第 7 章）。这些语言将出现在文档中的模式和描述链接结构（用路径正则表达式）的图查询结合起来。它们包括 W3QS[928]、WebSQL[72，1120]、WebLog[964] 和 WQL[1020]。第二代是 Web 数据操作语言，它们强调半结构化数据。然而，通过提供对网页结构（模型也包括内部结构）的访问，以及允许创建新的结构作为查询的结果，它们扩展了之前的语言。这个分类中的语言包括 STRUQL [555]、FLORID[767] 和 WebOPL[71]。这里提到的所有语言都是给程序使用的，而不是给最终用户使用的。然而，有些用于这些语言的查询接口的例子。

Web 查询语言已经扩展到其他 Web 任务上，如从网页上抽取和整合信息，以及构建和重建网站。关于 Web 查询语言的更多的细节可以在 Florescu、Levy 和 Mendelzon[569] 的一个很好的综述中看到。很多语言启发了基于 XML 的结构化文本的查询语言（见13.6 节）。

11.9.4 动态搜索

在动态搜索（有时叫做在线聚焦爬取，它可以看成 Web 中的序列搜索）的思想是动态地建立搜索语料库，在运行中通过跟随链接来发现相关的信息。主要的优点是允许用户搜索当前 Web 的“实时”结构，而不是由搜索引擎索引的文档集。换句话说，它相当于在运行时爬取。显然，出于可扩展性原因，这主要是为小的、动态 Web 子集而制定的，而不是为整个 Web 而制定的。第一个启发式设计是*鱼搜索*（fish search）[248]，它利用了这样的直觉，即相关的文档通常具有

相关的邻居。随后，被*鲨鱼搜索*（shark search）所改善［754］，它采用了之前在可视化搜索功能中提到的 Mapuccino/Fetuccino 工具，可以对邻接网页进行更好的相关性评估。相关的工作包括搜索 Web 特定信息的软件代理［1202，970］。这意味着需要处理那些必须被合并的异质信息源。在
503 这种情况下，重要的问题是如何确定相关的信息源（见第 10 章和第 17 章）。

11.10 相关问题

11.10.1 计算广告学

一个相关的 Web 搜索问题是计算广告学，这是“一个新的科学学科，是信息检索、机器学习、优化和微观经济学的交叉学科，它的主要挑战是在一个特定环境中找到最好的广告呈现给用户。这些环境包括在搜索引擎中查询（“广告搜索”）、阅读网页（“内容匹配”）、看电影和即时聊天等”［270］。该学科在 2009 年 9 月第一次成为斯坦福大学的课程。

计算广告学主要有两类。最有名的是为一个给定的查询匹配广告，在结果页面的右侧显示它们。这称为*广告搜索*（sponsored search），它是 Google Adwords 等系统的核心。第二种是为用户所请求的给定页面找到正确的广告。这种情况称为*上下文匹配*（contextual match），它是 Google Adsense 或 Yahoo Context Match 等系统的核心。

广告本身可以分为两类：图像或基于文本（或者两者的结合）。广告显示的初始模型主要是图像，而基于搜索的模型通常是基于文本的。在很多情况下，广告的放置通常被一个叫做*广告网络*（ad-network）的中介协调。商业模型通常是，对于显示广告的情况，按每次显示进行支付；或者通过广告搜索或者上下文匹配，按每次点击进行支付。另一种模式是按每次行动支付，如果特定的目标行为实现了，广告客户才会支付（例如出售产品）。

计算广告学可以看成一个搜索问题，其中搜索的输入，不管是查询还是页面内容，都必须与广告数据库进行匹配。数据库的每个广告至少有一个标题、一个目标 URL 和一个称为创意（creative）的描述，它在大多数情况下是文本形式。与经典的 Web 搜索的一个重要的区别是，在上下文匹配的情况中，搜索输入比广告本身要长很多。由于创意太短，与查询的纯文本的相似度（不管查询是短的还是长的，如在上下文搜索情况下）不会为用户带来足够相关的广告。为了解决这个问题，创意由大量关键词的列表系统性地增强，这或者由广告客户产生，或者在更复杂的情况下由广告系统提供。

匹配的广告必须排序，以便将最好的广告展示给用户。然而，这里对于“好”的衡量不仅基于相似度，而且还基于商业的考虑。事实上，在两种情况下，广告客户都通过用户在特定内容上的点击次数付费，直到他们预算所规定的限度。当预算支付完后，就不再显示更多的广告。这种花费被广告投标（和竞标）的拍卖机制建模，其标的是前面提到的与每个创意相关的关键字。在由 Goto 搜索引擎（后来更名为 Overture 并被雅虎收购）发明的最初方案中，这种顺序完全基于投标价格。然而，如果广告对于用户缺乏相关性或者没有价值，那么就不会有点击，模型就会失败。因此，主流搜索引擎实际采用的方法是应用拍卖竞标与相关性模型的结合。其中，拍卖竞标指的是广告客户对特定关键词的出价，而相关性模型则基于
504 用过去历史估计出的广告的期望*点击率*（Clickthrough Rate，CTR）进行预测。

还有一些研究工作致力于更好地理解广告的特征和用户点击它们的次数之间的关系，并观察如预期的那样，更相关的广告可以增加点击的数量。计算广告学是过于广泛和复杂（往往过于保密）的领域，在这里无法完全覆盖，但是接下来我们将介绍一些在这个领域上近期发布的研究结果。

Jones 等人 [850] 通过尝试将大量的查询（用户提交的查询集合）与非常小的广告列表语料库相匹配来解决这个问题。他们提出，为原始查询生成替代查询，以便拓宽可能的广告集合，然后对提出的查询排序。这是通过根据预先计算的查询与短语的相似度来修改部分的原始查询而实现的。数据从搜索引擎查询日志的用户会话中得到，因为大多数用户通过添加或删除词语来重构原始查询。通过结合一组特征，并生成一个能够最好地描述替代查询的组合模型，他们比较了一些机器学习技术。他们观察到，当所包含的词语的编辑距离改变比较小或变化比较少时，会得到更好的建议结果。

还有些方法用于解决上下文广告的问题。Ribeiro-Neto 等人 [1350] 描述了一个针对特定上下文广告的阻抗耦合（impedance coupling）技术，这可能是在这个问题上第一次发布的广泛深入的工作。他们关注于将有关的广告关键词与网页文本直接匹配的算法。他们的实验比较了 10 种不同的基于向量的排序函数，结果显示通过采用更复杂的排序函数，对于给定的页面能够显示出更好的广告。随后，lacerda 等人 [954] 探讨了应用遗传算法为上下文广告匹配学习排序函数（可能是对这个问题第一次应用学习技术）。他们显示排序函数和最好的阻抗耦合函数同样有效，并可以通过完全自动的方式产生。紧随其后的研究主要集中在从网页上抽取相关的关键词或短语，然后用来匹配描述广告的关键词。因为那些基于将网页的全部词语与广告关键词相匹配的方法（如刚才讨论的阻抗耦合算法）在实际中的计算代价比较昂贵，所以考虑到效率问题，这是非常重要的。

[1749] 提出了一种技术，该技术从文档中的短语和关键词中抽取一组特征，并且确定哪些对于目标广告是相关的。他们总共使用了 40 个特征，其中一些特征包括关键词是否是大写的、是否包含在标题中、是否在元数据中、是否在 URL 中、是否是名词等。他们也发现使用搜索引擎的查询日志中包含的查询是有用的，因为这些是人们所使用的关键词。所以，如果一个文档包含一些这样的短语，就可以用做文档的描述。他们从 MSN 搜索引擎中抽取了 750 万条英文查询。在从文档中收集了这些特征之后，他们对训练集中的相关关键词进行了手动分类，然后用一个监督学习方法来对未见过的文档进行分类。为了比较其方法的性能，他们应用 KEA[891] 作为基线，得到了更好的结果，不过这可能是由于他们使用了更多的特征。这种方法允许从内容中提取更多相关的关键词，从而获得更多相关的广告，增加整体的收入。 505

另一种方法由 Broder 等人提出的 [277]。他们不是应用描述性关键词来匹配短语，而是提出一个抽取语义和句法特征的系统，用于描述文本的内容，然后将它与广告匹配。他们使用由美国雅虎建立的一个分类体系，包括 6000 个描述查询的概念，其中每一个结点包括大约 100 个查询。为了使用这种分类体系对网页和广告分类，他们尝试了多种分类方法。最好的结果是通过将所有分在每个结点的查询串联起来，产生一个元文档。然后，他们应用这些元文档作为基于 Rocchio 分类器的最近邻分类的中心，并使用待分类文档和分类中心间的余弦距离，然后，他们为每一个网页设置一组类别（分类体系的主题）。为了得到特定网页的最终相关广告，他们结合了从分类中得到的语义信息（分类体系得分或 TaxScore）和句法特征（关键词得分或 KeywordScore），使用了两种得分的凸组合。

$$Score(p_i, a_i) = \alpha \times TaxScore(Tax(p_i), Tax(a_i)) + (1-\alpha) \times KeywordScore(p_i, a_i)$$

其中 p_i 和 a_i 分别对应于网页和广告，$Tax(x)$ 对应于从元素 x（网页或广告）的分类体系得到的类别集合。当两个元素处于相同的结点或者有相同的祖先时，$TaxScore$ 应该通过给出较高得分来反映广告和网页间的语义距离。$KeywordScore$ 通过将网页和广告在 n 维空间（每维对应一个项）中表示来得到，并计算向量间的余弦相似度。他们将这个语义-句法方法与纯句法方法进行了比较，并分析了 α 取哪个值可以得到更好的结果。从他们的观察可以推断出语义信息在匹配过程中是有用的，这是由于纯句法匹配只能依赖于页面的质量。

11.10.2 Web 挖掘

数据挖掘和信息检索之间一个基本不同是，数据挖掘必须在没有确切的查询或者信息需求的情况下发现信息。我们认为，正是由于这个原因，Web 挖掘超越了信息检索。Web 挖掘通常是在连续的三个阶段进行的，即数据的重新收集、信息提取和分析，旨在挖掘三种基本类型的数据，即内容、用途和结构。

1）内容数据包含文本和多媒体。

2）结构数据包含 Web 的链接结构（在更细的级别，可能也包括 XML 结构）。

3）用途数据包括 Web 日志、点击数据和用途访问模式（usage access pattern）。

此外，必须考虑一个正交的时间维，它反映了网页增长和演变的动态过程。因此，内容、用途和结构由时态数据补充。第一种和第三种类型在 [419] 中涉及，而第二种是 [349] 的主要主题。另一本通用的数据挖掘书籍是 [1037]。

内容挖掘可以细分为文本挖掘和多媒体挖掘。文本挖掘是一个经典的领域，超出了本书
506 的范围 [552，1706]，观点挖掘是现在流行的问题之一 [1240]。多媒体挖掘则更新，并且最近已经与其他情境（如地理和多样性）相结合。

链接挖掘是 Web 的内在问题，所以我们下面会提供更多的例子。ParaSite 系统 [1512] 使用超链接信息来发现那些已经迁移的网页、相关的网页和个人网页。HITS 也已经被用来发现社区和相似的网页 [625，911]。对超链接结构的其他探讨可以在 [352，1086，1273] 中找到。这个领域进一步的提高包括 Web 文档聚类 [269，361，1679]（已经提到过）、连接服务（例如询问哪个网页指向一个给定的网页 [197]）、链接自动生成 [671] 和信息提取 [226，261]。Web 垃圾的一些结果也被提到了，代表着链接挖掘的一种特殊情况（见 11.5.7 节）。

Web 用途挖掘是我们今天称为"群体智慧"[1546] 的一个最好的例子。Web 用途挖掘可以用做自适应 Web 设计（例如用户驱动的 Web 设计）、网站重组、网站个性化和某些性能的改进。与搜索相关的一类重要的 Web 用途挖掘是查询挖掘，接下来将详细介绍，它和搜索引擎也有内在的联系。

查询挖掘

最简单的查询挖掘直接与搜索用途相关，称为*搜索分析*（search analytics）。这是对与给定网站相关的搜索研究。它们包括来自搜索引擎的外部搜索和网站提供的搜索框中进行的内部搜索。第一种情况可以将只能通过搜索发现的页面与通过浏览发现的页面区分开来 [122]。仔细分析搜索，可以识别新的、更好的词来改善锚文本和网站组织。一个词比另一个词更能满足用户的信息需求，这个属性被 Pirolli[1268] 命名为*信息线索*（information scent）。内部搜索提供了在网站内部没有很好得到满足的信息需求，包括无点击结果或者空结果。这些说明网站拥有者可能没有意识到某个需求，同时提供了相关的关键词。甚至更重要的是，新的关键词预示了网站所缺失的新内容、新服务或者新产品的需求 [122]，给网站未来的发展带来了很多洞察力。事实上，查询可以比文档内容更好地描述文档 [1282]。其他有关利用查询来改善网站的应用在 [1383] 中涉及。

另一类主要的应用是使用查询挖掘来改善搜索引擎。我们在 7.2 节已经介绍了查询挖掘的很多例子，特别是关于意图、主题和模糊性预测。其他例子是 11.7.2 节介绍的查询推荐，以及 11.4.2 节陈述的基于查询的缓存和索引技术。Baeza-Yates 等人 [88，89] 和最近的 Silvestri [1481] 介绍了大多数挖掘查询日志以改善搜索引擎的应用程序。

然而，查询日志的另一个应用是语义关系的提取。通过分析提交的不同查询的用户行为（如点击数据），可以推断查询的语义，发现语义的相似度。Baeza-Yates 等人 [125] 分析了点

击图并获得有趣的语义关系。首先，对于日志上的每个查询，作为查询结果可以获得被点击的 URL 集合。这个 URL 的集合被称为 URL *覆盖*（UC_q）或者*点击图*。每个查询用 n 维空间内的结点表示，其中每一维是唯一的 URL，查询的每一维根据 URL 点击频率设置权重。随后，如果有相同的 URL，点击图中的结点（查询）会通过边相连，查询间的余弦相似度则作为边的权重。最后，每个结点都分配一个*带权的度*（weighted degree），它的值是与该结点相连的所有边的权重之和除以该结点的度。使用这种图，它们定义查询间的三种关系： 507

- **相同的覆盖**（$UC_{q1}=UC_{q2}$）：一条无向边，表示两个查询覆盖了同样的 URL，因此将它们定义成等价的查询。
- **严格完整覆盖**（$UC_{q1}\subset UC_{q2}$）：从 q_1 到 q_2 的有向边，代表 q_1 比 q_2 更具体这样的语义事实。
- **部分覆盖**（$UC_{q1}\cap UC_{q2}\neq\varnothing$，且不满足任何之前的条件）：这是最典型的情况，代表查询间部分相似。

使用这种查询表示和它们的覆盖图，就可以从中抽取语义信息。存在着一些包含多个主题内容的 URL，并对识别查询间的语义关系有着消极的影响。它们观察到低权重的边很可能代表较弱的语义关系，这就意味着与那些结点相关的 URL 可能是多主题的页面。利用这种观察从图中去除那些 URL，以减少由它们造成的噪声。根据分析结果，可以观察那些只通过单纯的语言方法不能确定的查询间关系，因为这种技术是语言独立的。例如，反复出现的一些错别字将产生 Web *俚语*（Web-slang），这只能从用途分析中确定。另外一个有趣的研究方向包括从查询建立一个层次式的分众分类法 [584]。

11.10.3 元搜索

元搜索是给多个搜索引擎、Web 目录和其他数据库发送特定查询的 Web 服务器，它收集答案并将它们组合到单一排序的列表中。它们可以看成是一种联合搜索，其中联合的来源是独立的搜索引擎（见 10.7 节）。元搜索在 Web 的早期是很流行的，那时搜索引擎还很少重叠。比较早的例子是 Metacrawler[1448]、SawySearch[511] 和 Vivisimo[1643]。大多数元搜索引擎已经消失或演变了（如 Vivisimo 转向企业搜索），所剩不多的元搜索引擎包括由 Vivisimo 支持的 Clusty[1]、Dogpile[2] 和自称为“搜索引擎之母”的 Mamma[3]。

在如何对组合结果列表进行排序（如果排序的话），以及如何将用户的查询转换成每个搜索引擎或 Web 目录特定的查询语言（共同的查询语言会很少）等方向，不同的元搜索各不相同。 508

元搜索的优点是结果可以通过不同的属性（如主机、关键词和日期）来排序，这些可以比单个搜索引擎的输出提供更多的信息。因此，浏览结果应该更简单。另一方面，结果不一定覆盖最相关的网页，因为每个搜索引擎返回结果的数量是受限的，较差来源的结果可能会被提升，损害了第二个搜索引擎产生的结果。这是只进行一遍处理的联合搜索的典型局限性。然而，直觉是通过多样化的来源，更多相关结果应该更容易被检索到。

在这个方向上最早的研究之一是 NEC 研究所的元搜索引擎 Inquirus[987，986]。Inquirus 下载并分析了从不同来源获得的每个网页，一旦它们变得可用，就以渐进的方式显示每个页面（突出显示所有的查询项）。

---

[1] http://www.clusty.com。
[2] http://www.dogpile.com。
[3] http://www.mamma.com。

元搜索的应用是合理的，通过早期的覆盖度研究表明，只有一小部分网页出现在所有搜索引擎中 [198]。事实上，这项早期研究表明，AltaVista、HotBot、Excite 和 Infoseek 索引的网页中只有不到 1%出现在所有的搜索引擎中。近期的研究则表明重叠加大了 [136，1518]，这可能部分地解释了为什么元搜索的流行性在缓慢地下降。企业搜索环境中的元搜索在 15.3.8 节中介绍。对于特定主题的元搜索可以认为是动态搜索软件代理，它在 11.9.4 中介绍。

对于元搜索主要的批评是，它们就像寄生虫，依靠在搜索引擎之上，没有昂贵的计算机基础设施的投资。出于这个原因，大型搜索引擎可能会限制它们每天从元搜索收到的查询量。

11.11 趋势和研究问题

考虑到 Web 从五年前才开始大量使用，它的未来可能会使我们感到惊讶。现在有很多不同的趋势，并且每一个都开辟了新的特定研究问题。下面将快速地回顾即将出现的数据源，它们应该会变得越来越容易获得，然后概括介绍更好的 Web 检索所面对的主要趋势和需要解决的挑战。

11.11.1 静态文本数据之外

这里我们认为更具挑战性的数据类型将不断出现，即暗（hidden）页面或动态页面、多媒体数据和语义数据。

1. 动态数据

相比于按需产生的内容，尤其是在查询电子商务或信息服务网站时，静态 Web 已经变得很小了。现在的爬取软件能够跟随动态链接，但是这必须谨慎地进行，因为动态生成的网
509 页在数量上是没有限制的。

访问查询表单背后的网页是更困难的，因为爬虫对于数据库没有先验的知识。这类网页组成了所谓的*深度网页*（deep Web）。另一方面，即使数据库是已知的，请求所有可能的查询可能太消耗时间（对于数据库的大小是指数级别的），即使我们只采用简单的查询，其中的某些查询也可能永远不会有人提出。如果允许 Web 服务（Web service）从数据库中学习，特别是学习人们如何查询，那么 Web 服务可能部分地解决了这个问题。例如，获取最频繁的 1 万个查询已经足够了。

2. 多媒体数据

多媒体数据包括：图像、动画、不同形式的音频和视频。它们都没有普遍认同的标准格式。图像的主要格式是 JPG、GIF 和 PNG，音乐是 MP3，视频是 Real Video 或者 Quicktime 等。理想的解决方案是使用同一个模型和单一的查询语言，搜索包括文本在内的任何类型的数据。这一雄心勃勃的目标可能无法实现。

对于某种特定的数据类型，我们能够开发一个相似度模型，根据其类型而改变查询语言。例如，通过示例的图像查询，或通过哼唱的音频查询（或者用 Shazam 模型[⊖]录音）。这个领域更多地属于图像和信号处理，而不是经典的信息检索。

搜索非文本对象在不久的将来将更加重要。目前已经有了很多的科研成果，将在第 14 章中讲到。

3. 语义数据

语义信息的两个主要问题描是描述语义的元数据标准和对于给定信息源的质量或信任

⊖ Shazam 是一个很酷的手机应用，在 iPhone 和 Android 上都可用；如果在手机上录制数秒钟长度的录音，它就可以通过 3G 连接识别歌曲。

度。第一个问题由万维网联盟处理，而第二个问题所要求的认证模式目前还未开发出来。

其他问题是一些常见问题，如扩展性、变化率、缺乏参照完整性（链接是物理的而不是逻辑的）、分布式授权、异质的内容和质量，以及多信息源。今天，一项主要的工作是开放链接数据（Open Linking Data）[1231]，它试图增加和完善 Web 上可用的语义资源间的链接。

语义搜索引擎代表着最新的发展。这些引擎搜索语义网数据。这种引擎最有趣的代表是 Sindice 引擎 [1483]，它可以搜索超过几千万个 RDF 文件，总共可能包含几十亿个三元组。[101] 讨论了一个更务实的语义搜索方法。Wolfram Alpha[⊖] 搜索引擎使用了一个完全不同的方法，它使用包含事实的知识数据库来推断出查询的答案。由于这个原因，它被称为答案引擎。 510

11.11.2 目前的挑战

Web 搜索是一个快速发展的研究和开发领域，我们列出了一些现有的挑战，它们需要付出更多的努力来解决。

- **分布式体系结构**。必须找出新的遍历和搜索 Web 的分布式方案来应对 Web 的增长。这将影响当前的爬取和索引技术，以及 Web 缓存技术。我们很想知道未来的瓶颈会在哪里，是服务器的容量还是网络带宽?
- **建模**。仍然需要更多为 Web 量身定做的信息检索模型。此外，搜索仍然主要是“拉”（pull）范式，即用户主动启动搜索；但为了更好的用户体验，“推”（push）范式仍然需要进一步地探索。在这两种情况下，我们需要更好的搜索模式和更好的信息过滤。
- **查询**。在查询中，需要进一步探索结构与内容的结合，以及新的查询和答案的可视化表示 [118]。未来的查询语言可能包括基于概念的搜索、自然语言处理，以及示例搜索（这意味着 Web 上的文档聚类和分类）。另一个关键问题是确定查询背后的需求：信息型、导航型或事务型，以及更细分的意图。根据统计，不到 50%的查询是第一类，这是典型的情况。一种替代的方法是为查询语言增加所需信息的环境，如类型或时间。因此为了更好地理解用户行为，需要广泛地研究查询日志。
- **排序**。需要更好的排序方案，充分利用内容和结构（网页内部和超链接）。特别地，结合或比较查询相关和查询无关的技术将会是很有趣的。一个与广告相关的问题是，搜索引擎可能将某些网页排在很高的位置，而不是基于网页的真实相关性（这就是在 [1086] 中所谓的搜索引擎说服（search engine persuasion）问题）。这也包含了更好地处理 Web 垃圾，以及识别高质量的内容。Web 充斥着低质量（句法和语义）的内容，包括噪声、不可靠和矛盾的数据，更不用说对可疑（恶意或非恶意）网站的信任问题。另一个挑战是为特定用户或一组用户提供相对应的排序，即围绕着个人或意图的个性化。相关性是基于个人的判断，所以基于用户轮廓或者基于用户上下文信息的排序可以有帮助。
- **索引**。尽管这个领域的研究和创新已经很久了，但还是有很多额外的问题需要解决。这些例子包括：文本的最佳逻辑视图是什么？应该索引什么？如何利用更好的文本压缩方案来实现快速搜索并降低网络流量？如何有效地压缩单词列表、URL 表，并在没有显著的运行时间损失的情况下更新它们？如何维护索引的新鲜度？很多实现细节必须解决和改善。 511
- **消除重复的数据**。我们需要更好的机制来检测和消除重复网页（或句法非常相似的网页）。最初的方法是基于使用文档指纹的类似性度量 [267, 269]，正如我们在

⊖ http://www.wolframalpha.com。

11.6.4 节中所看到的。这与数据库中寻找相似对象的重要问题相关。这个问题的一个变种可以处理通过搜索引擎发现的、受到排序影响的其他网页产生的内容 [120]。第 12 章覆盖了这个话题。

- **用户交互**。Web 搜索引擎的前端，无论是搜索矩形还是 SERP 范式，基本上是搜索引擎战争的前线，这对于在竞争中保留或者获取用户是决定性的。事实上，终端用户越来越难以评价相关性，他们的看法强烈地受到网页上用户体验的影响。常见的危险是塞满页面，落入到"多即是少"的陷阱，但这个领域还可以快速地继续发展。其他可探索的领域包括更好地抽取页面的主要内容，或者基于内容的查询描述 [1593]。另一个挑战是更好地使用用户的反馈，无论是通过显式的用户评价还是通过网络日志中的隐式反馈。
- **浏览**。这是一个值得重新审视的领域，可以利用链接、网页的流行度、内容相似性、协作、3 维和虚拟现实等技术，特别注意混合检索/浏览方法。
- **适应小规模语料库**。这是一个敏感的领域，尤其是对于不想开放内网的企业，以及仍然具有很少 Web 内容的新兴国家。由于语料库的规模有限，因此没有足够的内容/用途和链接数据来应用群体智慧的方法。这导致了在大规模的互联网比小的企业域更容易找到信息。
- **搜索时的内容传送**。搜索可以看做是在给定的时间内传送用户想要的内容这一普遍问题的一个十分特殊的情况。在搜索的情况下，这个动作是用户驱动的。另一方面，也可以是上下文驱动的（例如，Broder [272] 的信息提供范式）。因此，我们该如何通过设计所请求网页的内容来匹配当前基于用户的完整上下文（包括用户本身、历史、位置等）?
- **查询日志的隐私**。查询日志的使用对于搜索引擎是非常重要的。然而，在 2006 年的 AOL 事件[⊖]后，已经清楚地知道，有可能从查询分布的长尾部分中识别用户 [141]。即使我们不能识别一个用户，但年龄、性别或收入这样的信息也是隐私。出于这个原因，有些用户不喜欢被搜索引擎跟踪，例如他们在每个会话后删除 cookie。由于这个趋势，很多研究人员试图找到一些方式来和研究人员分享查询记录，但要
512 保持完全的匿名。然而对于非常罕见的查询，似乎并不可能。更多的细节见 Cooper 非常优秀的综述 [420]，以及在本章结尾的更多参考文献。
- **社交网络**。我们对于社交网络如何演变的理解仍然很少。此外，怎么能够将社交网络用做其他用途，如发现专家或社区、重要的路由信息等，都需要更多的研究。另一个重要的方面是社交网络和语义网的关系 [1128]。

11.12 文献讨论

有数百本关于 Web 的书。其中很多包括搜索 Web 和用户提示的一些信息。早期由 Abrams 编辑的一本书中包含了 Web 搜索的内容 [8]。其他较早的来源是《Scientific American》（科学美国人）杂志关于互联网的专刊（1997 年 3 月）和《IEEE Internet Computing》（IEEE 互联网技术）关于搜索技术的专刊（1998 年 7 月和 8 月）。

对 Web 建模的更多信息，建议查看 Baldi 等人的书籍 [130]。

最近涉及 Web 检索的书包括 Witten 等人的《Web Dragons》[1708]、由 Spink 和 Zim-

⊖ 参见 http://en.wikipedia.org/wiki/AOL_search_data_scandal。

mer 编辑的多学科文集［1522］和 Croft 等人的搜索引擎书籍［449］。关于分面搜索的更多信息可以在［1607］上找到，另外协同 Web 搜索可参见［1156］。

有关 Web 搜索的早期综述包括［64，268，348，745，747，1586］。链接分析方面的一个好的综述是［746］。信息线索的主题在［369，370，1269］中被扩展。关于幂律和无尺度网络的更多内容可以在［907，1142，1199，1341］中找到。关于 Web 长尾效应的影响，可以参见 Anderson 的书籍［50］。

关于缓存答案最新的比较参见 Gan 等人［617］。

对于高效地计算 PageRank 以及它的属性已经有许多研究了，可能多于真正需要的。这个领域的主要研究结果是［19，200，220，221，636，718，870，869，992，1111］。如果需要更多的细节，我们建议读者参阅 Langville 和 Meyer 的书籍［976］和 Berkhin 的综述［189］。

更多关于链接分析的信息可以在 Thelwall 的书籍［1575］和 Henzinger［746］的综述中看到。利用链接进行排序的其他算法包括［238，239，311，576，943，1200］。

如何优化一个简单的排序函数的很好的例子是 Singhal 等人的［1484］。

最近几年已经提出了一些排序学习技术。在点式方法中，我们有判别式信息检索模型［1170］和 MCRank［1019］。在对式方法中，我们可以查阅 RankBoost［591］、Ranking SVM［748］、RankNet［296］、IR-SVM［330］、FRank［1603］、MHR［1308］和 QBRank［1782］。在列式情况下，我们有 LamdaRank［295，507］、ListNet［331］、RankCosine［1307］、SVM-MAP［1759］、AdaRank［1733］和 SoftRank［1566，684］。Fen 等人研究了后者的损失函数［1726］。一个最近的综述来自 Liu［1064］。

从点击数据进行排序学习的工作可以参见［846，1322，1323］。学习也被应用到其他问题上，如 Web 搜索结果聚类［1768］、点击预测［18，1280］或者在线排序［1632］。总体来说，尽管只是部分地解决了这个问题，但“排序学习”已经成为信息检索领域一个重要的研究问题。尽管存在这些挑战，但这个领域是很有前途的，它能够将隐式和显式的用户信号 513
相结合，以便提高结果的质量。

许多研究人员都关注使用给定的类似度阈值来找到重复的文档，特别是 Garcia-Molina 及其合作者在数字图书馆环境下的工作［618，1465，1466，1467］。其他一些人关注于文档间的重叠［565，1154，1155］。I-match 算法的改进在［924，925］中讨论。

关于计算广告学的进一步研究参见［275，1319，1350］。更多的结果可以在 WWW 会议的 Monetization track 看到。

最近，一系列关于查询隐私的论文已经发表，特别是研究匿名技术［11，848，849，929，946，1728，1781］。Bar-Yossef 和 Gurovich 已经说明了，如何通过使用查询建议工具来估计查询频率和网页的效果［137，138］。查询日志的隐私问题已扩展成在网站环境下的商业隐私问题［1283］。隐私在社交网络中也是重要的［1781］。

此外，最好的 Web 文献来源是 Web 本身。刚开始，有很多网站对搜索引擎和 Web 目录专门进行告知和评级。其中，我们可以找到 Search Engine Watch［1543］和 Search Engine Showdown［1209］。关于 Web 特点的一个很好目录是［503］。提供 Web 搜索参考内容的其他来源有万维网联盟或 W3C（www.w3c.org）、万维网杂志（World Wide Web journal，w3j.com）和 WWW 会议系列（http://www.iw3c2.org/）。这些及其他参考内容在本书的网页中可以找到（见第 1 章）。 514

Web 爬取

——与 Carlos Castillo 合著

12.1 介绍

网络爬虫（Web Crawler，又称网络蜘蛛、网络机器人，或者简称为机器人）是一种从 Web 上自动下载网页的程序。在网页检索领域，爬虫抓取到的网页将用于索引和搜索（见第 11 章）。不同于其名字，网络爬虫并不像病毒或智能代理那样在 Web 上的机器之间转移，而是向其他地方的 Web 服务器发送文档请求。

让我们从一些具有历史意义的网络爬虫开始。1993 年 6 月，一名叫 Mattew Gray 的麻省理工学院（MIT）的本科生将下面这段消息发送到 **www-talk** 邮件列表中 [669]：

> 我写了一个 Perl 脚本，它在万维网上漫游，收集 URL，并跟踪它已访问过的地方和新发现的站点。最后，如果能创建一些代码，使它能返回一些更有用的信息（现在它只返回 URL），那么我将能生成一个关于这些网页的可搜索的索引。

当时这个项目叫做 WWWW（World Wide Web Wanderer)，后来成为第一个网络爬虫。它那时最多用于 Web 刻画研究 [668]，并且代表着一个重要的进步。为了说明这一点，第一个 Web 搜索引擎叫 ALIWEB（Archie-Like Index of the Web)，由 Martijn Koster 于 1993 年开发。它要求网站发布它们本地网页的部分索引，但只有少数网站会这么做。通过使用爬虫从网站上自动地收集网页，现代搜索引擎规避了这种不便利。

自然可以想到，自动爬虫将导致搜索引擎的产生。事实上，在 1994 年 6 月，华盛顿大学的一名叫 Brian Pinkerton 的博士生将下面这条消息发到了 **comp. infosystems. announce** 新
515 闻组 [1265]：

> WebCrawler 索引现在可以用于搜索了！索引的范围很广：它包含了来自尽可能多的不同站点的信息。它是个很棒的工具，在手动浏览时能帮助定位多个不同的起始点。当前的索引基于全球范围内接近 4000 台服务器上的文档内容。

然后，Web 规模持续快速地增长，从 1993 年到 1996 年，每年的网站数量均翻倍 [670]，最终导致了搜索引擎应运而生。而 WebCrawler 在变成一篇学位论文之前就取得了商业上的成功。基于网络爬虫的其他搜索引擎随后纷纷出现，包括 Lycos（1994 年)、Excite（1995 年)、AltaVista（1995 年）和 Hotbot（1996 年)。它们与 AOL、雅虎等公司提供的目录服务竞争。在当时，因为 Web 规模比较小，所以目录服务是一种更有效的搜索信息的方法。

如今，所有主要的搜索引擎都使用网络爬虫，它们的访问量占到网站所有访问量的 10%以上 [1205]。有效的网络爬虫是现代搜索引擎取得成功的关键。而且在很大程度上，“Web 之所以仍然对人类用户来说是易理解的，就是因为这些自动代理持续地分析和监测它们” [524]。

网络爬虫把多个种子网页作为输入，然后经过下载、分析和扫描等处理过程来获取新链接。对于指向未下载网页的链接，将它们加到一个中央 URL 队列中，用于后续处理。然后爬虫从队列中选择一个新的网页进行下载，这个过程不断重复，直到满足某个停止条件。一

个十分简单的爬取算法如图 12-1 所示。在 12.4 节，我们将描述网络爬虫的通用架构，在实际中爬虫要复杂得多。

```
Crawling( seed pages S )

URLQueue ← S
do {
    p ← Select-URL( URLQueue )
    content ← Download( p )
    (text, links, structure, ···) ← Parse( content )
    URLQueue ← Add-new-links( URLQueue, links )
    // Process text, structure, ··· depending on the application
} until( stop criterion )
```

图 12-1　一个十分简单的 Web 爬取算法

在接下来的章节中，我们将描述网络爬虫的可能应用，然后讨论爬虫的调度算法及其评价。最后，我们将介绍现阶段的趋势、研究课题以及相关的文献讨论。 516

12.2　网络爬虫的应用

网络爬虫最主要的应用就是建立一个包含宽泛主题（通用 Web 搜索）或者特定主题（垂直 Web 搜索）的索引。网络爬虫还用于对网页内容归档，以及自动地分析网站以抽取统计数据（Web 刻画）。对某些特定的网站来说，网络爬虫用来提高其设计（网站分析），或者保留它们网页的备份（Web 镜像）。

12.2.1　通用 Web 搜索

Web 搜索是近几年推动网络爬取技术发展的应用，大致可以分为通用 Web 搜索和垂直 Web 搜索，前者被各大搜索引擎所采用，后者限定在某一主题、某一国家或语言。Web 检索在第 11 章已详细介绍。

一个面向*通用 Web 搜索引擎*（general Web search）的爬虫必须注意平衡覆盖度（coverage）和质量（quality），其中*覆盖度*是指爬虫必须爬取到足够多的网页来回答很多不同的查询，*质量*是指爬取到的网页应该有高的质量。因为爬虫只有有限的资源，所以这两个目标可能会冲突。因此，爬虫需要根据一系列复杂的策略来工作，我们将在 12.5 节讨论它。

另一方面，一个面向*垂直 Web 搜索*（vertical Web search）的爬虫专注于 Web 的一个特定子集。可以从地理、语言或者主题等方面对这些目标网页子集进行定义。例如，有些流行的垂直爬虫应用程序建立了垂直门户（也叫做 vortals[⊖]），这些门户为特定用户的相关需求提供答案。

垂直爬取

垂直爬虫通常从不同的源头聚合数据，这些源头通常有相似或者相关的材料。垂直爬虫的一个最常见的形式就是*网店机器人*（shop-bot），它从在线商店目录下载信息，并提供一个界面来集中比较各个价格。另一个例子是*新闻爬虫*（news crawler），它从一系列事先定义好的源头收集新闻。它的主要目的是不断地爬取新闻，并频繁地更新这些信息，以保持其

⊖ http://www.wordspy.com/words/vortal.asp。

新鲜度。在生成这些网页集合后，这些新闻网页会被聚类以检测重复内容和相同主题，最后以聚合的形式展示给用户。

在 Web 上，同样存在一些带有恶意的爬虫，如为了得到嵌在网页中的电子邮件（E-mail）地址的爬虫，以便发送垃圾广告邮件到这些 E-mail 地址。这种类型的爬虫叫做垃圾机器人（spambot）。因为 E-mail 地址很容易识别，并且存在某些公共的邮件列表存档（这是 E-mail 地址的一个主要来源），所以垃圾机器人相对容易部署，也十分有效。对于垃圾机器人的一个对策是混淆 E-mail 地址，用术语来说就是“地址整理”（address munging）。
517 比如说，用一张图片来展示 E-mail 地址，或者对其编码。

特殊格式的垂直爬取

垂直检索也包括按照图片、音频或者视频对象等数据格式来分割的过程。在这类例子中，爬虫设计为只收集特定类型的对象，也就是说生成限定于某个特定数据格式的数据集。例如，CiteseerX[⊖] 以及其他文献服务所使用的爬虫通常仅仅索引 PDF、Postscript 和 Bibtex 格式的文件。同样，一个“feed 爬虫”（RDF 爬虫的一个特例）只检测网站上 RSS 或 RDF 格式文件的更新，这种类型的爬虫通常应用于网页新闻聚合器，以便周期性地检测一些预先设定好的网站更新。

12.2.2 聚焦爬取

如果爬虫设计为只对特定主题的网页或其他对象建立索引，那就可以把它叫做聚焦爬虫（focused crawler）。聚焦爬虫只爬取符合给定主题的网页，而不是爬取整个 Web。这是一个更加有效的策略，因为它避免爬取不必要的网页 [353]。

聚焦爬虫以对某个主题的描述作为输入，通常会用驱动查询（driving query）或者一系列已知属于该主题的样例文档来描述这个主题。聚焦爬虫的输出是与给定主题很有可能相关的更大规模网页的列表。爬虫能以批量模式工作，周期性地收集网页，或者以用户查询驱动按需收集网页。我们将在 12.5.1 节进一步讨论聚焦爬取。

12.2.3 Web 刻画

Web 刻画是组建有效 Web 搜索引擎的先决条件，它包括获取网页的所有统计属性。对于 Web 刻画，一般来说，最显而易见也是最困难的问题是，用什么来代表 Web。这是一个特别困难的问题，因为即使 Web 只包含有限的信息，它也可能包含无穷多的网页。

当网络爬虫用于 Web 刻画时，起始的种子网页和用于挑选新网页的爬取策略对刻画的结果有着很大的影响。在这方面，以网页为中心的刻画，由于其目的在于衡量页面大小、技术、标记和其他网页属性，所以要比以链接为中心的刻画受到更少的影响。对于以链接为中心的刻画来说，如何选择爬虫的起始 URL 是十分关键的，如果没有很好地选取种子 URL，那么观察到的 Web 的宏观结构可能就会有较大的偏向性 [1451]。

12.2.4 镜像

Web 上的镜像（mirroring）是指对某个网站保存部分或者全部副本的行为。镜像的目
518 的是为了分配服务器负载，给不同地方的用户提供更快访问速度，减少时延。这些副本通常

⊖ http://citeseerx.ist.psu.edu//。

叫做“镜像”。用于镜像的爬虫通常十分简单，它在一个受控的环境中运行，因此可以使用压缩和增量编码的方式更新镜像。当然，也存在镜像的策略，如合适的更新周期是多久，通常是每天或每周，再如一天中最佳的镜像时间是何时。

Web 归档（Web archiving）是指保留大量网页的镜像，即使是那些已经过期的副本。也就是说，每个网页的所有历史都会被记录。最大的 Web 归档项目是 Internet Archive（Internet档案库）[⊖]，它的目标是对 Web 上的所有公开的信息进行归档，包括文本和图片。2008 年 4 月的 Internet Archive 总共包含大约 850 亿个各种版本的网页，共计数百 TB 的数据。它的主要目的是在网页被删除或更新之前保留 Web 每年的状态。

12.2.5 网站分析

Web 爬虫可以用来分析网站，甚至根据预先确定的准则即时改变它（的行为）。自动网站分析最常见的形式是*链接验证*（link validation），即自动扫描网页，查找指向不存在网页的“坏链”。另一个常见的形式是*代码验证*（code validation），它保证所有的网页、样式表和脚本都符合各自所用语言的规范。一个更具体的例子是测试大型网站目录，它寻找那些已经无法访问的网站，并把它们列为目录中的待删除项目。

网站分析工具还可以用来查找网站的脆弱点，包括一些较旧的、未打补丁的流行脚本，它们可以用来取得未授权的服务器访问。网站管理员这么做的目的是加速和简化访问自己网站的流程，但不幸的是，这通常也是不怀好意的黑客所使用的伎俩。在大型文本资源库（如 Wikipedia）中，Web 爬虫能用来自动化很多任务，包括分类以保证所有同一集合下的网页符合同一标准，还可以检测未知版权状态的图片，或者检测*孤儿*（orphan）（未链接的）页面。

12.3 爬虫的分类体系

尽管所有类型的爬虫都希望提高覆盖度，越全越好，但它们在其他目标方面可能有所不同。比如，之前我们已经讨论过的爬虫在如下几个方面会有不同的侧重点：

- *新鲜度*（freshness）：在某些场合，保证爬虫得到的网页副本是最新的是很重要的；而在其他时候，即使是旧一点的网页副本也是能接受的。
- *质量*（quality）：有些爬虫的目标是部分特定的高质量网页，而另一些对广泛的覆盖度更有兴趣，尽管这些网页有不同的质量水平。 519
- *容量*（volume）：有些爬虫对保留大部分的 Web 网页有兴趣，而另一些可能会牺牲数量来换取更高的质量和新鲜度。爬虫的覆盖度依赖于该指标。

根据以上这三个维度对爬虫进行的分类，如图 12-2 所示。镜像系统需要保存一份非常精确而且完整的网页子集的副本，而垂直搜索引擎或者个性化爬虫优先考虑高质量的网页文档集，更强调内容部分。

网页类型

从爬虫的角度来看，可以用网页的类型将网页分类，它至少包括两个维度：公共/私有和静态/动态，如图 12-3 所示。

⊖ http://www.archive.org/。

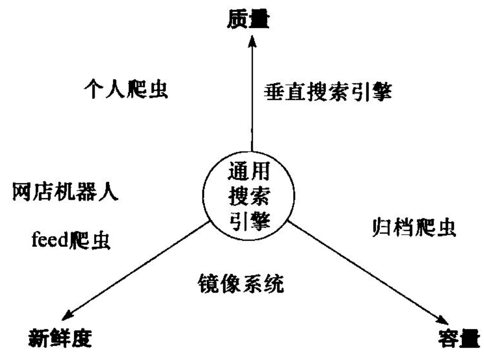


图 12-2 爬虫的类型

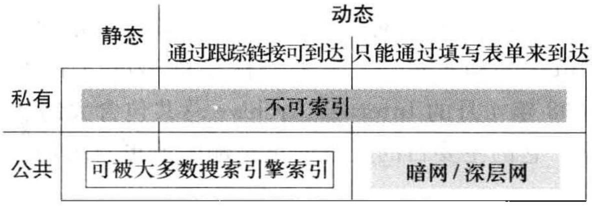


图 12-3 从爬虫角度看的网页类型

私有网页包括密码保护的网页，如通常内网中存在于防火墙后的网页。如今，私有网页还包括只有朋友、朋友的朋友，或者其他限制用户组才可见的社交网络数据。通常来说，所有这些网页都是不可索引、不可爬取的。另一方面，公共网页是那些原则上可以被网络爬虫索引的部分。

静态网页是那些在网站上等待用户访问的网页。动态网页是那些直到用户访问时才在 Web 服务器上产生的网页，它理解用户的请求，然后按需创建网页并返回给用户。动态网页的一个例子是搜索引擎的搜索结果页面，因为有大量可能的查询，不可能提前生成这些结
520 果页面。

在实际中，我们可以有无限多的动态网页（比如用软件生成的日历），所以我们无法期待爬虫能下载所有的动态网页。因此，大多数爬虫选择一个最大深度去爬取动态链接。实际上，[96] 说明了到 5 层深度就可以覆盖超过 90%的人们可能访问的网页。

不幸的是，并不是所有的动态网页都可以通过链接取得，仍然存在许多网页只能通过用户提交查询或者在线表单交互来得到。只有通过这种类型的交互才能到达的网页叫做暗网。尽管有一些方法能够索引它们中的某些（见 12.7.1 节），但现在的大部分爬虫还没有索引这部分网页。

12.4 架构和实现

正如我们将在 12.5 节所看到的，爬虫必须有一个好的爬取策略；但是它同样需要一个高度优化的架构。Shkapenyuk 和 Suel[1468] 说过：

> 尽管组建一个每秒只下载一些页面、运行较短时间的慢速爬虫是相当容易的，但是组建一个能运行数周、下载数以亿计网页的高性能系统，在系统设计、I/O 和网络效率、健壮性和可管理性等方面都面临着一系列的挑战。

尽管我们这里并不去描述高端的网络爬虫，但是我们将讨论它们的基本特征，并且提供足够多的细节以实现一个低端但还算实用的爬虫，我们将对图 12-1 所示的简单算法进行扩展。在 12.8 节，我们将涉及一些爬虫实现的例子。

12.4.1 爬虫架构

网络爬虫的典型高层架构如图 12-4 所示，爬虫由三个主要模块组成：下载器、存储器
521 和调度器。调度器（scheduler）是关键模块，它负责维护一个待访问的 URL 队列，也称为

“前沿”(frontier)，用于以特定的顺序将这些 URL 发送给一个或者多个下载器（downloader)。下载器负责获取每个 URL 所对应的网页内容，并解析给存储器（storage）模块以便后续索引和检索。另外，存储器模块也将获取的网页的元信息提供给调度器，这是用来驱动调度的重要策略。

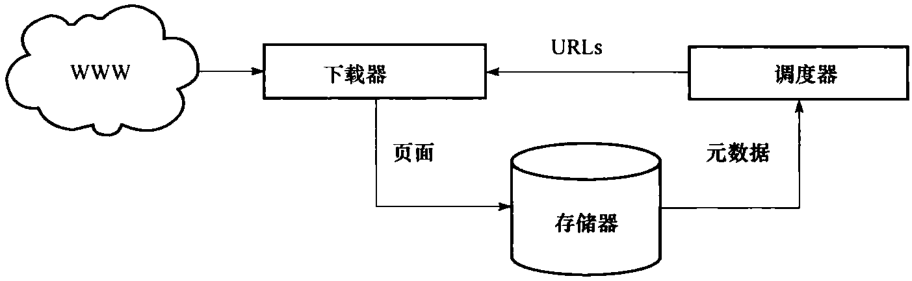


图 12-4 包含调度器和下载器的 Web 爬虫的高层架构

如果我们考虑调度可以进一步分为两个部分：长期调度（long-term scheduling）和短期调度（short-term scheduling)，那么这张图可以被细化，如图 12-5 所示。长期调度是指需要根据估计的网页质量和新鲜度决定接下来要访问的页面，短期调度是指为了遵从友好策略（见 12.5.3 节）或者优化网络使用必须重新安排网页访问顺序。长期调度器的时间尺度一般是几小时或者几天，而短期调度器的时间尺度是几秒到几分钟，这主要依赖于给爬虫配置的等待时间属性。存储器也可以进一步分为三个部分：文本（或者是保留一些或全部 HTML 标签的格式化富文本)、元数据（metadata）和链接(link)。

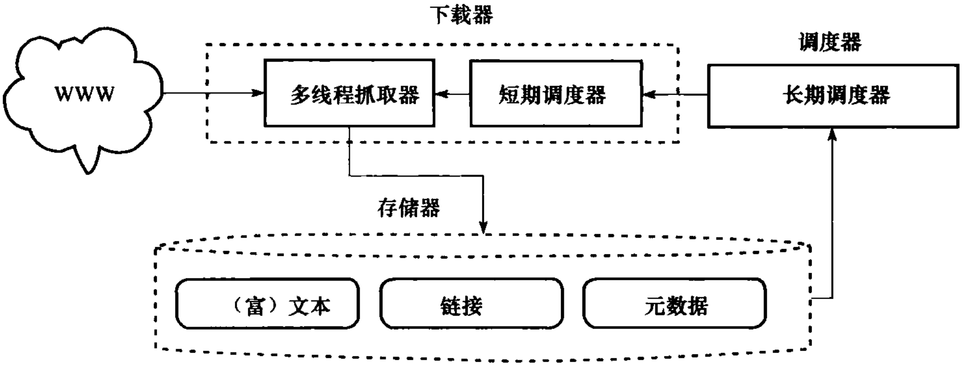


图 12-5 更具体的 Web 爬虫架构

为了达到调度目的，聚焦爬虫用文本信息来对网页分类，并优先下载某些 URL。通用爬虫使用元数据和链接信息来决定接下来要下载的网页。

对于短期调度器来说，执行友好策略需要维护多个队列，每个站点一个队列，每个队列中是需要下载的网页列表，如图 12-6 所示。在短期调度器中，有些线程可能会变“空闲”，如图 12-6 中的队列 1、3、5 和 7，这可能会成为效率不高的一个原因，我们将在 12.6 节讨论。当然，在我们所展示的通用架构基础上还存在很多变化，更多细节见 [349，第 2 章]。 522
还有许多实际的细节问题，比如 DNS 解析、（网页）解析、文本重复或近似重复检测和 URL 规范化，我们现在讨论它们。

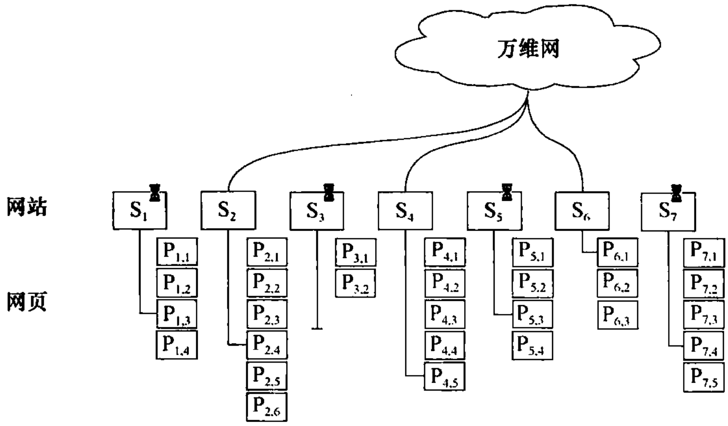


图 12-6 短期调度器的操作：每个网站一个队列，每次每个活跃队列最多一个连接（图中站点 2、4 和 6 是活跃的）

12.4.2 实际问题

除了带宽和存储能力外，爬虫的实现还涉及许多实际问题。大部分问题的出现都是因为爬虫需要与许多不同的系统交互，不同的系统会不同程度地符合标准和服务质量要求。

网络爬取最大的挑战之一是如何从多个来源下载网页，同时保证整体的输入流量尽可能均匀分布。但现实中 DNS 和 Web 服务器的响应时间变化很大，使得这个问题变得更加复杂。而且，Web 服务器的服务时间并不能随便假设，我们经常会看到 Web 服务器宕机很长时间，甚至几天或几周，然后又重新出现。

Henzinger 等人 [747] 认为从网络爬虫的角度来看，最重要的挑战是垃圾网页和重复内容。最近，Patterson[1246] 等人向 Web 搜索引擎开发人员推荐了一系列文章，解释了为什么 Web 搜索问题通常是比较困难的。最困难的两个问题是实现不符合标准，和高度普遍的重复内容。我们在这里将讨论每个问题，但在这之前，我们先讨论不同类型的网页、URL 规范化和（网页）解析。

1. DNS 解析

[301] 是最早的关于网络爬取研究之一，该作者挑选 DNS（域名服务）作为问题的主要源头之一。根据我们的经验，它包括临时 DNS 失效、不良或者错误的 DNS 记录，还有

523 DNS 解析的效率因素。

因为爬虫可能会使本地 DNS 满负荷，所以大多数爬虫会进行 DNS 缓存。也就是说，它们保存那些需要更频繁解析的域名所对应的 IP 地址。这种缓存即使在没有满负荷的时候也是重要的，因为从缓存中解析一个域名要比用标准的 DNS 解析效率高得多。

2. URL 规范化

Web 包含大量指向相同内容的 URL。一个典型的重复 URL 集合可能包含如下的字符串：http://x.example.com/、http://example.com/x/、http://example.com/x/index.html 和 http://example.com/x?sessionid=00000000，这些链接都指向同样的内容。尽管这些网页在下载之后可以通过某些方法（如 shingling[278]）检测出是否重复或者近似重复，但是通过对 URL 使用一些简单的句法规则，原则上有可能避免下载这些重复网页。

然而，大多数检测包含相似内容的 URL 的规则并不通用，许多是针对特定服务器的手动结果 [1442]。爬虫通常使用匹配流行网站的手写规则，比如，移除默认的文件名（如

index. html）和移除典型的会话标识参数（倾向于基于 cookie 的会话）。

可以使用自动过程来发现针对某个特定网站的规范化规则。在这个过程中，通过找到许多拥有相似内容的 URL 对，可以发现候选规则，然后使用一系列启发式的方法 [1442] 过滤这些候选（规则）。

3. （网页）解析

Web 上的很多网页的 HTML 代码比较差，并没有遵照 HTML 语言规范。其中一个原因是大多数 Web 浏览器有较高的容忍度，希望展示尽可能多的网页而不打断用户的体验。举例来说，如果一个网页的 HTML 代码有个错误，那么显示一个错误窗口会十分恼人，因为用户对此无法做什么。因此，相反地，即使 HTML 代码是错误的，浏览器无论如何都会渲染 HTML。

严格解析 HTML 几乎永远不可能，爬虫解析器模块必须允许 HTML 编码中有错误，即便是不明显的错误。同样道理，尝试给网页创建文档对象模型（Document Object Model，DOM）树在大多数情况下需要对网页预处理来改正编码错误（更多关于 DOM 的信息见 6.4.3 节）。

在解析过程中，信息抽取是十分重要的。这个过程可以是从标题和头部等简单的 HTML 标签抽取，也可以是复杂的自然语言处理。对于后者，主要过程之一是实体抽取。实体可以是名字（如人或机构）、日期及其他时间实体、地理位置（对本地搜索十分重要）。在这个阶段，我们也可以抽取属性-值对，特别是与结构化信息相关的。例如，在电子商务网站中，我们可以获得带有相关元数据（如价格、年份等）的产品信息。在这个主题上有很多的研究，见 [955，694，784] 的例子。 524

4. 软 404 页面

在忠于 Web 标准方面，HTTP 的实现各有不同。在很多情况下，对爬虫来说最具破坏性的问题是很难辨别一个 URL 是否存在。对于很多网站来说，如果爬虫尝试下载一个不存在的网页，那么服务器会返回一个指向某个自定义的错误页面的重定向，*并不会返回标志错误条件的响应头*（response head）。Bar-Yosef 等人 [1754] 将这些页面叫做“软 404”页面，并观察到大约 29%的死链指向它们。这个名字来自 HTTP 规范，其中网页不存在的错误编号是 404。

“软 404”页面对搜索引擎的爬虫是有害的，因为它们最终可能会被索引，并且它们通常不含有有用的内容。为了缓和这个问题的影响，有些爬虫先通过访问一些随机生成、几乎不可能存在的 URL 来测试该网站，然后验证它们是否得到正确的返回代码。如果网站没有返回“页面未找到”的错误，那么它们将以另外一种方式对待该 Web 服务器上的网页。而且，软 404 页面可以通过文本分类器自动地识别出来，其中文本分类器通过学习与这些网页的内容相关的一些短语或关键词得到。

5. 重复

Web 上的镜像内容十分多。1997 年，Border 等人 [278] 使用网页查重算法 shingling（见 6.5.3 节）估计出：1/3 的网页和另外某个网页几乎完全一致。其他人也有同样的观察结果，如 30%的网页是完全重复的 [379]，29%的网页是近似重复的，还有 22%的网页是完全一致的 [559]。这些重复中的某一些是有意的，对应于其他网页的镜像；而另一些重复是无意的，仅是由网站建设的方式造成的。实际上，最近关于 Web 演化的研究表明这部分的比例可能更高 [120]。

无意的重复（unintentional duplicates）有多个原因，主要的原因是嵌入在 URL 中的、

用于跟踪用户行为的标识（如/dir/page.html；jsessid＝09A89732）。这些标识用于检测逻辑会话。从爬虫的角度来看，这些会话标识是重复的重要来源，因为爬虫无法区分两个具有相同语义内容的网页。尽管如此，爬虫必须注意会话标识，并且在多个访问请求之间尽量都能保持会话标识一致。正如［531］所说的：“除非把这个先验知识告诉爬虫，否则仅仅在这一个网站上，爬虫就基本上能够找到无穷多的 URL 来爬取”。即便这些重复网页可以通过查看其内容检测出来，但在理想情况下爬虫应该避免下载它们，以免浪费网络资源。

一个相关的问题是 Web 上信息的*颗粒度*（granularity）。博客、论坛和邮件列表存档都是典型的大型资源库，它们由个人用户的很多小帖子组成。当主题在其他地方并未涉及时，
525 它们是很好的信息源。典型的例子是技术支持消息，它们通常简短地描述十分具体问题的解决方案。然而，有时候个人帖子没有其他网页有价值，因为其长度十分短。一个单独的帖子只包含少量的信息，但是整个对话可能很有价值。爬虫可能会参考网站建设的方式来只索引那些聚合信息的网页（对话线索），而避免索引每一个单独的帖子。

12.4.3 并行爬取

为了得到更好的可扩展性和容错能力，网络爬取必须是多线程的；对于大规模文档集，它应该是分布式的。

在爬取时有必要采取多线程的方式，因为爬虫可用的带宽通常会比单个网站的带宽要大得多。多线程意味着爬虫请求下一个网页之前不需要等待某个下载线程结束。

当在分布式的环境中运行爬虫时，最重要的问题是避免多次下载同样一个网页，或者过载 Web 服务器。进程之间通过交换 URL 来协调运行，设计爬虫时的目标是最小化（进程间）通信开销。理想情况下，每个网页应该只由单个进程下载。

一个完全分布式的爬取系统需要某个策略来分配新发现的 URL，因为发现新 URL 的进程不一定是下载它的进程。由哪个进程来下载给定 URL 是由*分配函数*（assignment function）决定的，所有的进程从一开始就知道这个函数。已知大多数链接都指向同一网站上的网页，所以这个分配函数应该把整个主机（host）分配给同一个进程。散列函数可以用来将主机名转换为数字，这个数字对应于相应爬取进程的索引。

Boldi 等人［219］说道，有效的分配函数必须有三个主要属性：每个爬取进程应该得到大致相同数量的主机（平衡属性）；如果爬取进程的数量增加了，那么分配给每个进程的主机数量必须减少（逆变属性）；分配函数必须能够动态地增加或者移除爬取进程。他们提议使用*一致性散列*（consistent hashing），它复制*散列桶*（hashing bucket），所以增加或移除一个桶不需要重新散列整个表来获取所有需要的属性。有了这个属性，新的进程可以进入爬取系统，而不必重新散列所有主机。应用这样的分配函数，没有页面会被爬取两次，除非某个爬取进程垮掉了。在那样的情况下，另一个进程必须重新爬取来自*失败代理*（failing agent）的网页。爬虫 UbiCrawler［219］（Java 语言）实现了这些想法。

最后，有一些启发式的方法来减少由于交换 URL 造成的开销。在爬取进程之间批量交换 URL（每次多个 URL）十分重要。如果所有进程在爬取之前就知道文档集中大部分被引
526 用的 URL（如使用上一次爬取的数据），那么它能帮助减少进程间通信［375］。

12.5 调度算法

Web 爬虫需要同时平衡各种各样的对象，其中一些对象还会互相矛盾。一般来说，

Web 爬虫需要网页的新鲜拷贝，所以它必须重新访问网页来探测变化。同时，它又必须发现新网页。它必须高效地利用网络带宽，如下载“好”的网页以避免浪费资源。然而，爬虫并不能提前知道哪些网页是“好”的，它必须先下载它们。

更麻烦的是，Web 是动态的，也就是说，每天都有大量的网页增加、更改或删除。较高的变化率说明，当爬虫刚从数据集中下载完最后几个网页时，很有可能新的网页已经出现了；或者已经下载的网页已经被更新，甚至删除了。

某种程度上，爬取网页与在晴朗的夜晚看天空类似：我们所看到的星星的位置反映不同时间星星的状态，因为光传播了不同的距离才到达我们。爬虫所得到的并不是 Web 的真正快照，因为在任何确定的瞬间，爬虫所收集到的网页集并不代表当前的这个 Web。最近爬取的网页可能是十分准确的，但是早些时候爬取的网页很有可能已经过时了。这个比喻如图 12-7所示。

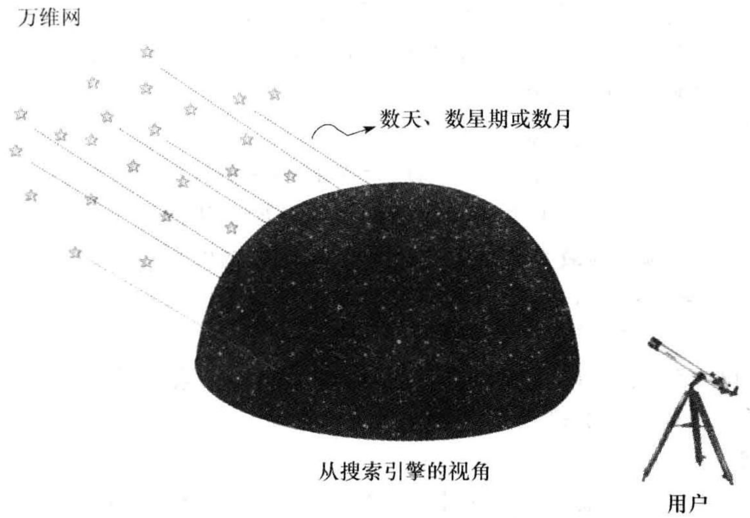


图 12-7 从搜索引擎视角，Web 代表着不同时间的网页状态

最简单的爬取调度算法是以宽度优先的策略遍历网站。以这样的方式，我们不仅能够提高网站的覆盖率，而且不会连续访问一个网站的许多网页，有助于遵守友好策略。然而这还不够，对于一个在大数据集上运行的 Web 爬虫，如全球规模的搜索引擎的爬虫，比较有用的是将爬虫的行为看做一系列独立策略的组合。例如爬取算法可以看成下面三个不同策略的组合，每个算法都有各自不同的目标： 527

- *选择策略*：其目标是优先访问质量最好的网页。
- *重访问策略*：其目标是当网页改变时更新索引。
- *友好策略*：其目标是避免过载网站。

12.5.1 选择策略

在现有的 Web 规模下，即使是大型搜索引擎也只能覆盖公开可访问内容中的一部分。Lawrence 和 Giles[989] 在 2000 年的一项早期研究表明，在当时没有任何搜索引擎能够索引超过 16%的网页。一份更近的 2005 年的报告 [685] 表明，大型搜索引擎的网页覆盖率在 58%～76%。因为爬虫总是只下载网页的一部分，所以很希望下载的那部分网页包含了最具权威性的网页，而不只是 Web 的随机采样。正如 Edwards 等人 [525] 提到的：

> 由于用于爬取的带宽既不是无限的，也不是免费的，所以如果为了保持合理的质量和新鲜度指标，爬取网页的方式必须不仅具备可扩展性，而且还要是高效的。

爬虫必须在每一步都小心地选择接下来要访问的网页。例如，回到 2004 年，与一次大规模 Web 爬取相关的带宽花费估计达到惊人的 150 万美元 [438]。尽管当前的带宽花费往往更少，但是整个 Web 变得越来越大。也就是说，一次完整的大型爬取将仍然花费很大，这说明高效使用网络资源来最大化爬虫吞吐量以及避免浪费已分配的带宽是十分重要的。

选择哪些网页来爬取可以分为如下两种类型：*离线限制*（off-line limit），它是预先设置的；*在线选择*（on-line selection），它是在爬虫运行时计算的。后者包括聚焦爬虫，它的选择标准是爬取与某个特定主题相关的网页。

1. 离线限制

在实际中，由于存储的限制，经常需要为爬取进程预先设立限制，这些限制可以是整体上的最大爬取数量。然而，由于网站和主机之间的幂律分布（见 11.3.3 节），因此以每个主机或域的形式表示这些限制更有用。

较多使用的网络爬虫离线限制如下：

- 爬取的最大主机数量。这对于大的域之下的垂直爬取特别有趣。
- 528 最大深度。例如从任何主页或者起始网页集开始遍历的最大链接数。
- 数据集中所有网页的最大数量。它依赖于存储数据的可用空间。
- 每个主机或域的限制，包括最大网页数或者从每个服务器下载的最大字节数。
- 接受的 MIME 类型列表（如 text/html 和 text/plain）。

同时，每个网页的限制可能包括网页大小的最大值、只索引某些网页的开始几个词语，以及限制每个网页所要处理的最大外链数。

2. 在线选择

在线选择要爬取的网页需要有网页重要性的指标，使得网页有优先度。一个网页的重要性可以是它的内在质量、体现为链接或访问数的流行度，甚至其 URL 的函数（后者如垂直搜索引擎限制爬取某个顶层域，或者搜索引擎限定于某些网站）。设计一个好的选择策略有其他一些难点，它必须处理部分信息，因为在爬取时爬虫并不知道完整的网页集合。

Cho 等人 [378] 最先进行了关于爬取调度策略方面的研究。他们的数据集是在 stanford.edu 域的 18 万个网页，在这上面他们测试了不同策略的模拟爬取。测试的排序指标有广度优先、反向链接计数和文章随后定义的局部 PageRank。测试得出的结论之一是，如果爬虫在爬取时优先下载 PageRank 值大的网页，那么局部 PageRank 的策略更好，其次是广度优先和反向链接计数。

Najork 和 Wiener[1169] 使用广度优先的策略在不同域的 3.28 亿个网页上进行真实的爬取。他们发现广度优先的爬虫会优先爬取 PageRank 值较大的网页（但他们并没有与其他策略比较）。对此结果，作者给出的解释是，最重要的网页有来自众多主机的许多链接指向他们，而且这些链接会被很早发现，无论爬取最先是从哪个主机或者网页开始的。

Abiteboul 等人 [6] 基于在线页面重要度计算（On-line Page Importance Computation，OPIC）算法设计了一个爬取策略。在 OPIC 算法中，每个网页都被赋予了一个初始数目的“现金”，它被平均分配给该网页指向的页面。这类似于 PageRank 计算方法，但是它更快，而且只做一步。基于 OPIC 的爬虫优先下载那些在爬取前沿中有较多“现金”的网页。实验在一张具有 10 万个网页、入链服从幂律分布的合成图上进行。结果显示，作为指导爬虫的

指标，OPIC 方法比简单的入链计数方法要好。

Boldi 等人 [220] 在来自 .it 域的 4000 万个网页和 WebBase 的 1 亿个网页上进行模拟爬取，测试广度优先策略与随机排序策略和全知策略。最后获胜的是广度优先策略，尽管随机排序策略也表现得非常好。出现的问题是 WebBase 上的爬取偏向用于收集数据的爬虫。他们也发现，在 Web 的部分子图上计算 PageRank 是真实 PageRank 值的一种较差的估计，而且如果这些部分的计算应用到爬取策略上时，好的网页更不容易出现。 529

Baeza-Yates 等人 [100] 在 Web 的两个子集上进行模拟，分别是来自 .gr 和 .cl 域的 300 万个网页，测试了不同的爬取策略。他们发现 OPIC 策略和一种使用每个网站的 URL 队列长度的策略均比广度优先策略要好。而且，即便当 Web 快速变化时，使用上一次爬取收集的信息来指导本次爬取也是十分有效的。

3. 聚焦爬取

在线选择策略的一个特定例子是根据给定的主题过滤网页。主题描述通常由驱动查询组成，有时候也可以是一个样例文档集合。如果给聚焦爬虫提供了样例文档，那么这些文档同样可以用做启动爬虫的种子网页。聚焦爬虫与通用爬虫以同样的方式运作，但不是把所有未见过的 URL 都加到队列中，而是只加入那些可能与提供的主题相关的网页。与主题的相关程度可以由所有可用的信息推断出来，包括驱动查询、样例文档和到目前为止聚焦爬虫见过的网页。

聚焦爬取利用 Web 上的*主题局部性*（topical locality）。这意味着相互链接的网页比随机选择的网页更有可能属于同样的主题 [477, 1119]。特别地，相关的网页往往被共同引用，也就是说如果网页 A 链接网页 B 和 C，而且网页 B 与网页 A 的主题相关，那么很有可能网页 C 也与同样的主题相关。随着更多的网页被爬取，垂直爬虫的专注度会提高，利用这个有趣的反馈效果，可以很高效地识别并忽略噪声网页。

聚焦爬取所要解决的主要问题是，从带宽利用率的角度来看，在实际下载网页之前就预测给定网页文本与查询间的相似度会更高效。一个可能的预测器是链接的锚文本，这正是 Pinkerton[1266] 所采用的方法。Diligenti 等人 [497] 提出了一个基于"上下文图"的方法，其中那些导向相似网页的网页的全部内容可用于推测目前还未访问过的网页的相关度。

聚焦爬取的性能极大地依赖于搜索的特定主题或类别的链接丰富程度。例如，学术网页往往比对应的商业网站包含更多更好的链接 [1576]。这是因为商业网站几乎不会链接竞争对手的网站。

12.5.2 重访问策略

Web 具有动态的天性，爬取 Web 的一部分可能会花费很长时间，通常需要数周或数个月。在爬虫完成爬取工作时，许多事件可能已经发生了。我们把这些事件描述为创建、更新和删除 [98]： 530

- **创建**：当创建一个页面时，它并不是立即在公共 Web 上可见，因为没有链接指向它。所以，至少需要某个网页更新后，增加一个链接指向那个新网页，新网页才可访问，故而可见。一旦网页变成可见，它就可以根据重访问策略被爬取到。
- **更新**：网页变化很难描述，但基本上，更新要么是小的，要么是大的。如果更新发生在段落或者句子级别，那么我们说它是小的，所以网页在语义上几乎还是一样的，指向该内容的引用也仍然有效。相反，如果更新比较大，那么所有指向该内容的引

用就失效了。如何描述部分变化在 [1031, 1215] 中有研究。

- **删除**：如果一个网页无法再被访问，那么可以称之为被删除了。需要注意的是，即使所有指向某个网页的链接都被删除了，并且网页在 Web 上也不再可见，但 Web 爬虫仍然能够访问该网页，只要知道网页的确切 URL。但几乎不可能去检测一个网页是否丢失了所有指向它的链接，因为爬虫无法判断指向该网页的链接是否都存在，或者这些链接只存在于还未被爬取的网页。未检测出的网页删除对搜索引擎的名誉损害比更新要严重，因为他们对用户来说更明显。Lawrence 和 Giles 关于搜索引擎性能的研究 [989] 报告，平均 5.3%的搜索引擎返回的链接所指向的网页已经被删除了。在搜索引擎服务器保存的本地缓存缓和了这个问题。

1. 与网页事件相关的代价函数

从搜索引擎的角度来看，未能检测出一个事件就需要付出代价，因此有过时的网页拷贝。较常使用的代价函数是新鲜度和年龄（age）[374]，我们现在讨论它们。

*新鲜度*是一个二进制的指标，表示本地的网页拷贝是否是最新的。

资源库中网页 p 在时刻 t 时的新鲜度定义如下：

$$F_p(t) = \begin{cases} 1 & \text{如果 } p \text{ 和时刻 } t \text{ 的本地拷贝相同} \\ 0 & \text{否则} \end{cases} \tag{12-1}$$

*年龄*用于衡量本地网页拷贝的过时程度。资源库中网页 p 在时刻 t 的年龄定义如下：

$$A_p(t) = \begin{cases} 0 & \text{如果 } p \text{ 自最近一次更新后未更改} \\ t - t_{last\ update} & \text{否则} \end{cases} \tag{12-2}$$

新鲜度和年龄随着时间的变化趋势如图 12-8 所示。

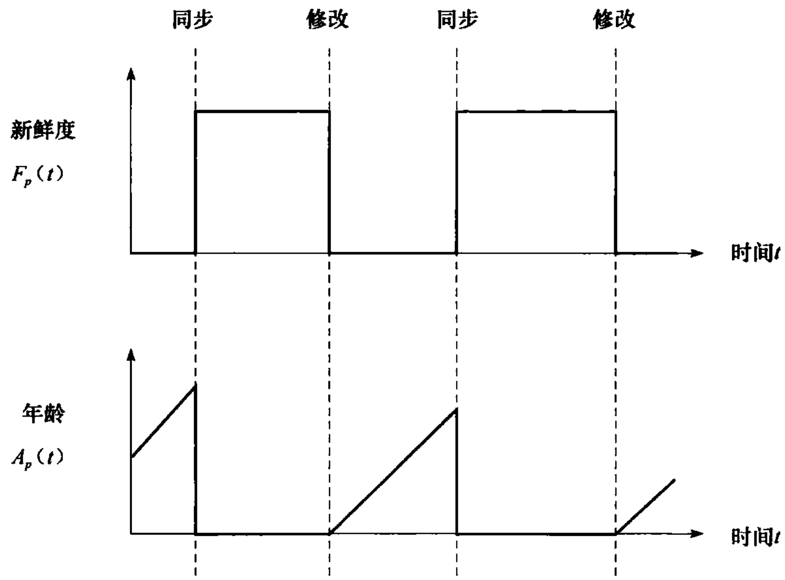


图 12-8 新鲜度和年龄随时间的变化 [374]。两种事件可能发生：修改服务器上的网页（修改事件）和爬虫下载修改后的网页（同步事件）

保持资源库新鲜的问题，从排队论角度可以建模成一个多队列、单服务器的排队系统 [404]。用排队论中的术语来说，爬虫就是服务器，每个网站是一队等待服务的客户。网页
531 修改和客户到达每个队列类似。在这样的模型下，客户的平均等待时间是需要最小化的变量。这个变量与爬虫对所爬取网页的平均年龄设置有关。

另一个不同的方法是，如 Wolf 等人 [1712] 所建议的，考虑给最终用户提供的服务质量。对于爬虫来说，作者提出了一个*尴尬度*（embarrassment）指标，它对应于一个过时的网页被查找结果的用户点击的概率。因此，为了决定更新哪些网页，爬虫需要从搜索引擎的查询日志中抽取相关信息。

2. 策略

爬虫的目标可以是保持本地文档集中网页的平均新鲜度越高越好，或者保持文档集中网页的平均年龄越低越好。这两个目标不是等价的。前者，爬虫仅仅关心有多少网页是过时的；后者，爬虫关心网页的本地拷贝有多旧。

Cho 和 Garcia-Molina[376] 提出了一个模型，模型中所有的网页从质量角度来说都是同等重要的，他们考察了两种重访问策略：*均匀的*（uniform）和*按比例的*（proportional）。*均匀策略*是以同样的频率重访问文档集中的所有网页，不管它们变化的速率如何。*按比例策略*更经常地重访问那些变化较频繁的网页，在这样的情况下，访问频率直接与网页的估计变化频率成比例。

在这两种情况下，网页的重复爬取能够以随机或者固定顺序的方式进行。Cho 和 Garcia-Molina 证明了一个惊人的结果，在平均新鲜度方面，*均匀策略*无论在模拟的 Web 还是真 532 实的 Web 爬取上表现得均更好。这个结果的解释是，当一个网页变化过于频繁时，爬虫将浪费时间在快速重新爬取上，却仍然无法保持网页拷贝的新鲜度："为了提高新鲜度，我们应该惩罚那些太频繁改变的网页。" [376]

无论均匀策略还是按比例策略都不是最优的。保持较高平均新鲜度的最优方法包括忽略变化过于频繁的网页，保持较低平均年龄的最优方法是随着每个网页的变化频率，单调（且次线性）递增爬虫的访问频率。在这两种情况下，均匀策略要比按比例策略更接近最优方法。正如 Coffman 等人 [404] 所说的："为了最小化期望的过时时间，对任何特定网页的访问要尽可能平均。"

显式的重访问策略公式通常来说无法得到，因为它们依赖于网页变化（频率）的分布。如果这个分布是已知的，那么最优化的重访问策略可以数值化地表示出来。例如，[525] 描述的爬虫使用了非线性规划的方法来解决线性系统中最大化期望新鲜度的问题。

通常来说，大多数大型搜索引擎会使用两三个队列，它们的更新时间各不相同：一个队列用于新闻网站，每天刷新多次；一个队列用于流行或相关网站，每天或每周更新一次；还有一个大型队列用于其他的网页，它每月或者每几个月刷新一次。

需要注意的是，这里所考虑的重访问策略认为所有网页在质量方面是同类的，也就是说 Web 上的所有网页都有相同的价值——当然这并不现实。所以，需要关于网页质量的更多信息，以得到更好的爬取策略，这点会在 12.5.4 节讨论。

3. 新鲜度估计

对于每个网页 p 来说，下列信息在每次访问之后就可知了：

- 网页 p 的访问时间戳：visit_p。
- 网页 p 最后一次修改的时间戳（由大部分 Web 服务器提供，在实际中，在大概 80%～90%的访问请求中可以得到）：modified_p。
- 网页的文本，它可以用来与更旧的拷贝比较，以检测网页变化，特别是当 modified_p 未提供时。

如果重访问的周期足够短，那么可以估计下面的信息：

- 网页第一次出现的时间：created_p。

- 网页无法再访问的时间：deleted_p。Koehler[917] 在一个持续数年的长期研究中说到，许多当时无法访问的网页在未来又可以访问了。

在所有的情况中，得到的结果只是真实值的估计，因为它们是通过对事件（变化）轮询得到的，而不是通过事件通知，就是说，网页可能在连续的两次轮询之间发生了变化。如果
533 网页 p 未被访问，那么 p 的拷贝在时刻 t 是最新的概率 $u_p(t)$ 随着时间的增加而减小。Brewington 和 Cybenko[258] 认为，如果某个网页的变化以独立的时间间隔发生，那么就可以建模成泊松过程。然而，值得注意的是，大部分网页变化表现出一定的周期性，因为大部分更新是在与所研究网页样本相关时区的工作时间发生的。所以那些没有考虑这个周期性的估计器在周或月的尺度上比在较小的尺度上会更有效。

网页变化可以建模成一个泊松过程。在这种情况下，如果自上次访问之后，又过去了 t 单位时间，那么：

$$u_p(t) \propto e^{-\lambda_p t} \tag{12-3}$$

参数 λ_p 是网页 p 的平均变化率，它可以根据先前的访问来估计。

假设我们已经访问了一个网页 N_p 次，其中 X_p 次访问观察到网页变化。现在 S_p 是自第一次访问网页已经过去的时间。一个直接估计 λ_p 的方法是：

$$\lambda_p \approx \frac{X_p}{S_p} \tag{12-4}$$

然而，这个估计是有偏的，主要是因为爬虫所观察到网页的变化次数比真实发生的变化次数要小。这个估计还有其他一些问题，Cho 和 Garcia-Molina[377] 进行了分析。他们提出并分析了一个更好的估计器，该估计器还考虑到多数 Web 服务器对每个网页都提供一个“最后修改”的时间戳。为了计算这个估计值，在每次访问的时候，爬虫必须按如下方式累加从网页变化开始的总时间 T_p。如果爬虫访问时并没有发现网页被修改，那么从上次访问到现在的时间被加到累加器上；如果访问时网页已经被更改了，那么从服务器提供的上次修改时间到现在的时间被加到累加器上。然后，λ_p 可以估计为：

$$\lambda_p \approx \frac{(X_p - 1) - \dfrac{X_p}{N_p \log(1 - X_p/N_p)}}{T_p} \tag{12-5}$$

对于这一变化频率估计器及其他估计器的细节可以参看 [377]。

4. **网页变化的刻画**

尽管对网页有不同的时间相关的指标，最常用的是：

- 年龄：$\text{visit}_p\text{-modified}_p$。
- 生存时间：$\text{deleted}_p\text{-created}_p$。
- 生存时间内变化的次数：changes_p。
- 534 平均变化间隔：$\text{lifespan}_p/\text{changes}_p$。

一旦从样本网页中得到上述值的估计，就可以计算出整个样本的一些有用的指标，比如变化间隔的分布、网页平均生存时间，以及网页的中位生存时间，即 50% 的网页发生变化所用的时间，也可以叫做 Web 的半衰期——这是从物理学里借来的词汇。挑选出来的一些关于网页变化的结果在表 12-1 中进行了总结。不幸的是，研究这些参数的方法是各种各样的，因此很少有可比较的结果。有些研究人员关注网页的生存时间，因为他们关心 Web 内容的可用性。另一些研究人员关注网页变化的频率，因为它与网络爬取更直接相关，知道了这一点可以帮助生成好的重访问顺序。

表 12-1 关于网页变化的部分结果（按样本大小排序）

参考文献	样本	观察
[917]	360 个随机网页，长期研究	半衰期约为 2 年 33%的网页维持 6 年
[1092]	500 个网页（联机论文）	半衰期约为 4 年半
[689]	2500 个网页，大学网站	平均寿命约为 50 天 中位数年龄约为 150 天
[1514]	4200 个网页（联机论文）	半衰期约为 4 年
[373]	720 000 个网页，流行站点	平均寿命约为 60～240 天 40%的 .com 网页每天都变化 50%的 .edu 和 .gov 网页在 4 个月内保持不变
[509]	950 000 个网页	平均年龄在 10 天～10 个月 有很多链接的网页变化得更频繁
[1215]	400 万个网页，流行站点	每周有 8%的新网页 62%的新网页有新的内容 每周有 25%的新链接 80%的网页变化都是很小的
[561]	1.5 亿个网页	65%的网页在 10 周时间内保持不变 30%的网页只有小变化 在不同域间可访问性有很大不同
[258]	8 亿个网页	平均寿命约为 140 天

从学术研究的角度来看，研究网页的过时也是十分重要的课题，因为引用在线发表的资源正变得越来越常见。大多数人假设这些在线资源是持久的，但实际上它们并不是。为了解决这个问题，**数字对象标识符**（Digital Object Identifier，D. O. I.，见 http://www.doi.org/）目前被大部分主要的科学文献出版者所采用。否则，经常需要报告一个引用的 URL 最后一次被看到的日期。

12.5.3 友好策略

如 Koster[934] 所说的，Web 机器人的使用尽管对某些任务是有用的，但对一般社区来说却是需要付出代价的。网络爬虫需要可观的带宽，并需要在较长的时间内高度并行操作，所以它们消耗了过多的带宽。这造成服务器过载加重，特别是当对服务器的访问频率过高，或者机器人的代码没写好时。 535

这不是唯一的忧虑，隐私也同样是 Web 爬虫的一个问题。比如，它们可能访问网站并不打算公开的那部分内容，索引这些内容，然后公开给用户访问。最后，如果操作机器人的搜索引擎对下载的网页保留了一份缓存，目前还未实施的版权问题可能会出现。

有些准则对爬虫的持续执行同样十分重要。如果考虑到多个网站会托管在某些物理服务器上，那么对某个网站不友好的爬虫可能会被托管提供者的所有网站所禁止，从而无法爬取许多网站的内容。

成为社区共识、所有主要搜索引擎都遵守的、关于爬虫操作的三条基本规则是：

1）Web 爬虫必须声明自己是爬虫，不能假装自己是正常的 Web 用户。这样做的目的包括，可以正确计算网站的用户访问数，但同时又使得在某些情况下分配给爬虫的带宽可见和可控。

2）Web 爬虫必须遵守机器人排除协议（robots exclusion protocol）[933，935]，它为管理者明确说明了服务器的哪些部分不能被机器人访问。

3）对于某个 Web 服务器，爬虫必须保持低带宽使用。这意味着，爬虫不应该从同样的网站上同时下载多于一个网页，而且在连续两次下载之间必须等待一段时间。

例如，四个较大的搜索引擎遵循如下规则：

- Ask 搜索：http://about.ask.com/en/docs/about/webmasters.shtml
- 谷歌（Googlebot）：http://www.google.com/webmasters/bot.html
- MSN 搜索：http://www.msnsearch.com/msnbot.htm
- 雅虎搜索（Slurp!）：http://help.yahoo.com/help/us/ysearch/slurp/

下面，我们将详细讨论关于机器人识别、排除协议和带宽使用等问题。

1. 机器人识别

大多数分析 Web 流量的软件都有方法区分网民和爬虫。这是有意义的，因为大多数网站运营者都想要精确地统计他们网站的独立访问人数，而爬虫的访问可能会人为地提高这个数字。即使 Web 服务器可以检测到爬虫的浏览模式 [1557]，但是如果爬虫能够声明自己的
536 身份，那么将会变得更有效得多。

联系人信息对网站管理员来说也十分重要。即使当爬虫是友好的，并采取保护措施防止 Web 服务器过载，但还是会出现问题，导致（用户）抱怨。Brin 和 Page[263] 说道：

> *……运行一个连接超过 50 万台服务器的爬虫……产生相当多的邮件和电话。由于大量的用户刚开始上网，总有人不知道爬虫是什么，因为这是他们见到的第一个。*

HTTP 协议 [564] 包含一个 **user-agent** 字段，它可以用于识别是谁发出了这次访问。来自爬虫访问请求的 user-agent 字段中应当包括某个网页的地址，该网页包含爬虫的信息、邮件地址或联系信息。如果没有提供这些信息，那么网站管理员可能会投诉整个来源网段的所有者。

大多数 Web 流量分析软件包含大型搜索引擎的爬虫所使用的 user-agent 描述。对于小型爬虫来说，建议在这个字段中包含“crawler”或“robot”这个词，以便正确地识别身份。

2. 机器人排除协议

机器人排除协议 [933，935] 涉及三种类型的排除：服务器范围（server-wide）、网页级（page-wise）排除和缓存的排除。

*服务器范围*排除规则告诉爬虫哪些目录不应该爬取。这是通过位于服务器根目录中的 `robots.txt` 文件实现的。文件的语法十分简单，基本上每行一条指令，每条指令说明哪些 user-agent（即爬虫的名字）必须遵守所列出的限制，如不能下载的目录。例如：

```
User-agent: *
Disallow: /data/private
Disallow: /cgi-bin
```

上面这段说明，所有爬虫都不能下载目录 `/data/private` 和 `/cgi/bin`。

*网页级*排除规则通过在网页中包含元标签（meta-tag）来实现。元标签是标准 HTML 语法的一部分，它允许网页作者在网页中加入 `key=value` 形式的多对信息。例如，下面的元标签：

```
<meta name="robots" content="noindex,nofollow"/>
```

它表示爬虫既不能索引这个网页，也不能跟踪这个网页中的链接。

*缓存排除*通常由销售信息访问权的发布者使用。尽管他们允许爬虫索引全部网页的内容，以保证其网页链接可以存在于搜索结果中，但他们告诉搜索引擎不要将本地缓存的网页的拷贝展示给用户。这是通过在与网页级排除相同的 HTML 标签中，使用 **nocache** 关键词实现的：

537
```
<meta name="robots" content="nocache"/>
```

即使这些预防措施都做到了，爬虫还是可能会访问那些不希望公开的网页，结果可能会导致法律问题。因此，从爬取的网页数据库中快速删除某个文档的方法十分重要。

3. 控制带宽使用

爬虫可用的带宽通常比它访问的网站的带宽要大得多。如果使用多个线程，爬虫可以轻松地使网站过载，特别是小网站。为了避免这个问题，通常对一个网站只打开一个连接。而且，在连续的两次访问之间，爬虫会延迟一段时间。这个延迟时间是一个重要的参数，[933] 第一次提出了 60 秒的建议。然而，如果以这样的速度从一个网站下载 10 万个网页，那么即使存在零时延且带宽无限的完美连接，也要花两个月的时间才能全部下载完。此外，我们会长久地使用 Web 服务器的部分资源，这点可能也是无法接受的。

Cho[377] 建议使用 10 秒作为连续两次访问之间的时间间隔，而 WIRE 爬虫 [95] 使用 15 秒作为默认值。Mercator 爬虫 [759] 则遵循一个自适应的友好策略：如果从给定的服务器下载一个文档要花 t 秒，那么爬虫会等待 $10\times t$ 秒再下载下一个网页。Dill 等人 [498] 使用 1 秒的时间间隔。来自访问日志的一些坊间证据显示，知名爬虫的访问间隔从 20 秒到三四分钟不等。

最近，有些 Web 爬虫（其中有雅虎、MSN 搜索和 Ask）已经开始遵从某个扩展的机器人排除协议，它允许网站运营者决定爬虫在索引他们的网站时所用的延迟时间。只要在 **robots.txt** 文件中加入一行来指定 crawl-delay 的值即可实现它，例如：

```
Crawl-delay: 45
```

它告诉爬虫，在连续两次访问之间要等待 45 秒。

12.5.4　组合策略

为了组合我们已经介绍过的策略，我们首先要说明的是，友好策略只对爬虫的短期行为有影响。友好策略的时间尺度比选择和新鲜度策略的要小。我们可以利用这两种不同的时间尺度把爬虫的行为分为两部分：短期调度和长期调度，前者处理友好策略，后者处理选择和新鲜度，如 12.5 节所述。

例如，考虑到我们有网页的质量和新鲜度的估计，一个很自然的组合是考虑下载单个网页所得到的收益（profit）[95]。假设本地文档集中的一个网页的质量估计值是 q（如基于全局的排序顺序），它保持最新的概率是 p（如基于新鲜度估计），那么我们可以认为当前它在索引中的价值是 $p\times q$。如果我们现在下载这个网页，那么它保持最新的概率变成 1，而其价值就变成 q。于是，下载该网页的期望收益是 $q\times(1-p)$。一个很自然的策略就是，按照期望收益对网页排序，优先下载最大化平均期望收益的网页集合。 538

需要注意的是，其他类型的衰减指标可以用于在资源库中“不新鲜的”网页。例如，现有指标值可以扩展为 $q\times p^{\alpha}$，其中参数 α 使得新鲜度比质量更重要或者相反。

12.6　评价

本节将讨论如何评价爬取策略。实际上，公平地比较不同的爬虫是不可能的，因为一个公平的比较应该让爬虫在同样的网站上运行，而且需要在严格相同的网络和服务器条件下。

12.6.1　评价网络使用

图 12-9a 展示了对于一个假想的、由五个网页组成的批过程最优的爬取场景。x 轴是时间（如以秒为单位），y 轴是速度（如以字节/秒为单位），所以每个网页的面积是它的大小。

下载器的最大带宽是 B^*，所以爬取工作可在如下时间完成：

$$T^* = \frac{\sum_i \text{size}(P_i)}{B^*} \tag{12-6}$$

其中 size(P_i) 是网页 P_i 的大小。

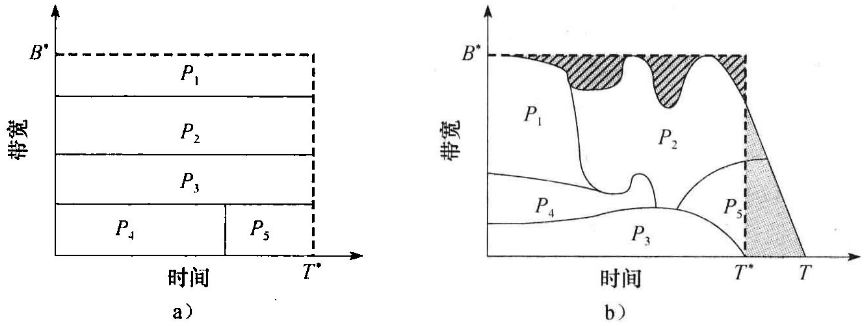


图 12-9 a）最优的短期调度，b）实际的情况。因为下载网页的总大小仍然是一样的，所以灰色区域和斜线区域的面积相同

让我们考虑更实际一点的情况：

- 下载每个网页的速度是可变的，所以在某些点，部分带宽很有可能浪费了。注意，爬虫的带宽比它所访问的网站的最大速度要高。
- 同一网站的两个网页不能一个接着另外一个立即下载，必须执行友好策略。

在这样的假设下，可能会出现不同的爬取时间线，如图 12-9b 所示。在最优的情况下，使用最大带宽 B^*，爬取会在时刻 T^* 时结束；而在实际情况下，需要更多的时间 T，因为某些过载或者远距离服务器的带宽被浪费了。浪费的带宽可以用 $B^* \times (T-T^*)$ 来度量。

如果一批网页在网页数量和主机数量之间服从幂律分布（也就是说，有少数大站点和许多小站点），那么当快要结束时很有可能只有部分主机是活跃的。如果最后只有部分主机是活跃的，那么一旦大部分已经下载完了，爬取就应该停止。尤其是，如果剩下的主机数量十分少，那么带宽就不可能完全被使用。

通过使用更多的线程，短期调度可以使用满带宽，减少完成时间。然而，如果使用了太
539 多的线程，那么在线程之间切换控制的代价就变得很昂贵。通过在处理过程的任何时候都避免只有少数几个网站可供选择，网页的访问顺序也可以优化。

12.6.2 评价长期调度

在 12.5.1 节，我们说明了长期调度的几个策略。为了比较不同的策略，[378，220，100] 以及其他人使用下面这些思想。首先，对文档集中的每个网页计算一个估计网页质量的指标。然后，对于每种策略，将网页按照下载顺序排列，并对每个网页计算所选质量指标值的总和。这样的实验结果是类似于图 12-10（来自 [100]，使用来自 **.gr** 域的 350 万个网页的文档集）的一张图。图中使用了 PageRank 作为质量指标，模拟了多个爬取策略，所有这些策略都从一个任意、但固定的结点集合开始。

图 12-10 模拟的爬取策略有三个：入度（in-degree）、广度优先（breath-first）和全知（omniscient）。广度优先策略之前已经提到过，它实现了一个先进先出队列来保存新发现的 URL，并使用先进先出的准则来调度 URL 下载。入度策略通过计算被所有已下载网页所指向的次数来选择接下来下载的网页。

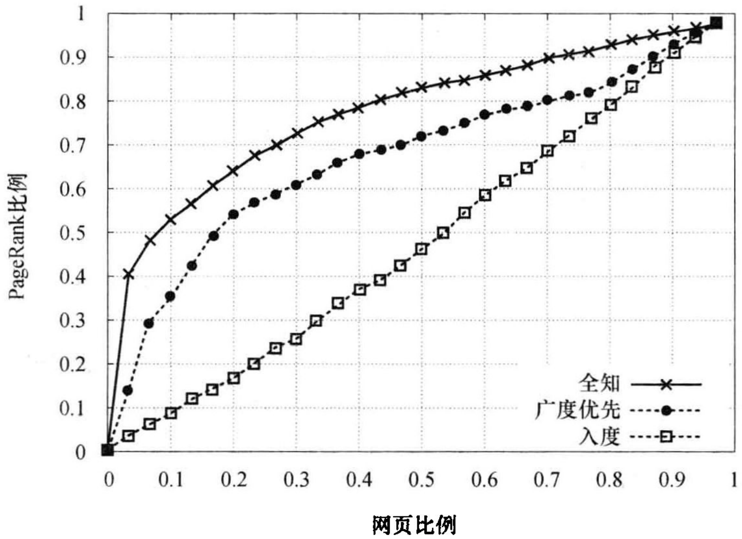


图 12-10 采用 PageRank 的长期调度评价

全知是一个理想中的策略，它查询某个已经知道整个 Web 图，并已经事先计算出每个网页的真实 PageRank 的“预言者”（oracle）。当这个策略需要优先下载时，它会向预言者查询排名最高的网页并下载它。注意，这个策略和其他策略一样，受制于某些限制，只能下载那些被已下载的网页指向的网页。这代表了任何可行策略的性能上限。

如果把 PageRage 作为质量指标，那么图 12-10 说明全知策略比其他两个策略表现得都好。注意整个爬虫的概貌可以通过单个数值来衡量，例如在爬取期间质量指标的平均值（曲线下面积），或者在某个固定点的质量指标值（如在爬取 10%时），它们根据使用爬虫的特定应用而定。 540

12.7 趋势和研究问题

到目前为止，我们已经讨论了一些标准的爬取技术。本节将讨论对基本爬取范式的一些扩展，以及一些研究挑战。

12.7.1 爬取“暗网”

我们所讲的过程可以索引所有通过跟踪链接可达的网页。Raghavan 和 Garcia-Molina [1324] 发现，在 Web 上，仍然存在大量公开可访问的信息，但是无法通过跟踪链接到达，只能通过查询或者提交表单，如图 12-3 所示。这部分 Web 称为“暗的”（hidden）或“深的”（deep）[259]；曾有估计，在 2000 年 [259]，这类网页的规模（网页数）多达整个 Web 的 550 倍。新近的研究表明，暗网的规模可能要更大 [357]。

为了爬取包含暗网网页的数据库，首先解析常规爬取所发现的网页，在其中查找表单；然后把这些表单转换成内部形式，对于每个表单项，它应该包含一个或多个标签；再把这些标签与内部的特定领域的资源进行匹配；当匹配完成后，生成并执行一个或多个特定领域的查询，然后由爬虫解析并保存查询的结果。

例如，一个收集旅行信息的爬虫可能有很多流行的旅行目的地。这样的爬虫在它找到一个输入地名的表单时，能够自动地发送查询。这些表单可以通过查看网页中的字段名或者它们周围的文本识别出来。 541

12.7.2 在网站帮助下的爬取

到目前为止，我们已经知道，对爬取来说，Web 服务器是被动的，但是也不尽然。有趣的是，对 1993 年那个描述第一个 Web 爬虫的帖子的回复已经提出了这种可能性 [1421]：

> 如果你能问每个服务器能否连接，这样是不是更好呢？这样似乎会使得事情运作得更快。因为每个服务器都能访问本地的所有信息，所以它能够快速、唯一地找到所有的 HREF 引用，并报告给其他服务器。

实际上，第一个搜索引擎 ALIWEB 就采用了这个思想。Web 服务器必须在一个本地文件中公布它们的内容，然后由爬虫组合各种各样的本地文件。不幸的是，大多数网站开发人员都太懒了，并未实现这些系统。大多数的现代爬虫都把 Web 服务器当做被动的对象。

从搜索引擎的角度来看，或许最重要的问题是检测网页是否改变了。目前，这主要还是通过轮询（polling）来实现，但把这种方式变成某种通知（notification）的形式或许更高效，正如在现代计算机中使用设备中断，而不是轮询。

对于给 Web 爬虫推送最后修改数据的最具体的提案由 Gupta 和 Campbell[687] 提出。它包含一个代价模型，在这个模型下，只有当网站被搜索引擎错误展示的程度超过一定阈值时，才会向爬虫发送元数据。

如今，Web 爬虫从多数网站那里得到的唯一帮助与同样能帮助普通用户的一些技术相关。这包括用于确认网页是否改变的"if-modified-since"请求，它是 Web 代理常使用的技术，对 Web 爬虫也十分有用。这还包括压缩，特别是对大规模的爬取十分有用。

对于实际使用的通知方案，使用最多的是在 RSS（Really Simple Syndication）的 ping 服务，如 http:/www.rssping.com/，它是博客和新闻源所使用的方案，用来向搜索引擎和新闻聚合器通知更新。实际上，RSS ping 的工作是，当加入了新的内容或者旧的内容被修改时，让新闻聚合器重新爬取由内容提供商发布的资源描述框架（Resource Description Framework，RDF）。

Web 爬取还存在许多其他有挑战性的主题，如：

- 改进选择策略，如开发一些策略，使得在爬取初期就能够根据已经收集的信息发现相关的网页。现在，在能够得到的最好的网页排序与最佳的启发性方法之间还存在差距。在 Web 图中高质量网页聚合（或分散）的程度很重要。
- 提高内存和网络使用率，特别是对于内存受限的环境，如桌面计算机的按需爬取。
- 542 爬取以获得事实，包括爬取语义网和聚焦爬取。
- 在其他环境中实施爬取，例如在对等服务中。
- 爬取社交网络，以及实时 Web，如 Twitter。

12.7.3 分布式爬取

另一个重要的趋势是从不同的地理位置爬取，也就是分布式爬虫。因为网络拓扑也有可能成为瓶颈，所以这也是重要的。在这种情况下，我们可以小心地将 Web 爬虫分布到不同的地理位置上 [538]。这个优化问题有很多变数，包括在不同位置的网络使用代价和把数据发送回搜索引擎的代价。在最近的一篇论文中，Cambazoglu 等人 [325] 显示，如果采用分布式技术，不仅能够更快爬取，而且我们能够优先爬取更相关的网页。

12.8 文献讨论

1994 年提出第一个爬虫架构 [529，1103，1266]，当时 Web 上只有几十万个主机。关

于其他早期爬虫和软件代理的细节见 [368, 1698]。

后续的、有较大影响力的有关爬取的论文包括 Najork 和 Wiener 关于广度优先爬取的工作 [1169], Cho 和 Garcia-Molina 发表的系列论文 [373, 374, 375, 376, 377, 378, 379], 以及 Chakrabarti 在聚焦爬取方面的开创性工作 [353]。最近的综述有 Olston 和 Najork[1229]。其他的好综述包括 Chakrabarti 的数据挖掘书中关于爬取的章节 [349], 以及 Castillo 的博士论文 [342]。

现在, 我们提供一些关于不同类型爬虫的线索。

(1) 全球范围的爬虫

Internet Archive (Internet 档案库) [301] 使用了一个叫 "Heritrix" 的爬虫, 设计目的是为了对 Web 的大部分保存周期性的快照。它以分布式方式使用了多个进程, 将一些固定数量的网站分配给每个进程。隔一段较长的时间才按序进行进程间 URL 交换, 因为这是十分耗时的。Internet Archive 的爬虫也需要处理改变 DNS 记录的问题, 所以它保留一份主机名到 IP 映射的历史档案。

谷歌的早期架构在 [263] 中详细描述, 包括爬虫是用 C++和 Python 编写等细节。爬虫和索引进程进行了整合, 因为文本解析既用于全文索引又用于 URL 抽取。有一个 URL 服务器用于发送 URL 列表, 它们将被多个爬取进程处理。在解析时, 发现的 URL 被传送到 URL 服务器, 以查看该 URL 是否之前已经被看过了。如果没有, 将这个 URL 加到 URL 服务器的队列中。

FAST 搜索引擎的架构在 [1361] 中有所描述。这是一个分布式的架构, 其中每个机器有一个 "文档调度器" 维护一个文档队列, 其中的文档将被文档处理器下载并保存在本地存储子系统中。每个爬虫通过交换超链接信息的分发器 (distributor) 模块和其他爬虫进行交流。 543

(2) 模块化爬虫

Mercator[759] 是一个用 Java 编写的模块化 (Modular) Web 爬虫。它的模块性来自于它使用了可交换的协议模块和处理模块。协议模块与如何获取网页相关 (如通过 HTTP), 而处理模块与如何处理网页相关。标准的处理模块仅仅解析网页, 然后抽取新的 URL, 但是其他处理模块可以用于索引网页的文本, 或者从 Web 收集统计数据。

WebFountain[525] 是一个与 Mercator 类似的分布式模块化爬虫, 但是它是用 C++编写的。它的特色在于它有一个控制机器来协调一系列 "蚂蚁" (ant) 机器。它还包含一个用于最大化新鲜度的方程组描述和求解模块。

WebSPHINX[1134, 1135] 是一个 Java 类库, 它实现了多线程网页检索、HTML 解析, 以及一个设置初始 URL 的图形用户界面, 用于抽取下载的数据, 从而实现一个基本的文本搜索引擎。

(3) 开源爬虫

NUTCH[1217] 是一个用 Java 实现的爬虫, 它是 Lucene 搜索引擎的一部分, 目前由 Apache 基金会赞助。它包含一个用于内网网页爬取的简单界面, 以及一系列更为强大的用于大规模爬取的命令。

WIRE[95, 343] 是一个用 C++编写的 Web 爬虫, 它包含多种调度网页下载的策略和一个生成关于所下载网页的报告和数据的模块。正因为后者, 它已经用于 Web 刻画。

ht://Dig[783] (用 C 编写) 是一个用于域或内网的索引和检索系统, 它包含一个 Web 爬虫。

其他在文献中提到的爬虫还有 WebBase[768] (C 语言)、CobWeb[472] (Perl 语言)、PolyBot[1468] (C++和 Python 语言), 以及 WebRace[1745] (Java 语言)。 544

第 13 章

结构化文本检索

——与 Mounia Lalmas 合著

13.1 介绍

文本文档往往显示结构信息，例如，科研论文包含逻辑结构，如摘要、多个章节和小节，每一部分包含多个段落。书包含布局结构，例如页面、分栏等。其他文本文档则可能包含由词性标注器输出的结果。这些结构信息通常使用标记语言（参见 6.4 节）进行编码，如 SGML 格式，以及现在主要使用的可扩展标记语言（XML）格式。

信息检索的很多阶段都可以利用结构信息。首先，在索引阶段，可以利用它识别和索引文档的不同部分，作为独立但相关的单元；其次，在检索阶段，可以利用它对文档不同粒度的组成部分进行检索；再次，在结果展示阶段，也可以利用结构信息选取文档的一部分进行显示，而不是文档的全部；最后，查询语言可用于在查询阶段形成要检索的内容和单元结构约束。

考虑这样一位具有很好记忆力的用户，他回想起了一篇感兴趣的文档中包含的一个页面，在这个页面中“red wine”出现在一个“figure”（图）的周围，这个图的标签中包含“Chile”这个单词。如果可以只返回这个页面，那么用户一定十分高兴。利用经典的信息检索模型，查询可以按如下方式表达：

```
"red wine" and "chile"
```

使用这样的查询会返回包含这两个字符串的所有文档。显然，结果中一定包含了很多用户不想看到的文档。在这个特定的情况下，用户更愿意使用更加丰富的语言来表示查询，例如：

545

```
same-page (near ("{red wine}", figure (label ("chile"))))
```

这个查询表达了详细的视觉回忆。这个例子说明准许指向文本内容和附着在文本上的结构约束的检索语言的吸引力。

将文本内容信息和文档结构信息进行结合的检索模型叫做*结构化文本检索*（structured text retrieval）模型。从 20 世纪 80 年代后期到 20 世纪 90 年代中期，提出了许多结构化文本检索模型。在 20 世纪 90 年代后期，随着 XML（1998 年）的提出，越来越多的人开始关注结构化文本检索的研究。XML 现在也成为一种结构化文档的事实（de-facto）标准。对于 XML 检索的研究，在 2002 年 INEX 启动之后有了进一步推动。INEX XML *检索评测*（Initiative for the Evaluation of XML Retrieval，INEX）[⊖]准许研究人员比较和讨论那些为 XML 检索所特别研究的模型的性能。现在，XML 检索几乎是结构化文本检索的同义词。

本章在 13.2 节讨论由结构化文本检索模型带来的多种类型的结构化能力。13.3 节将介绍早期的结构化文本检索方法，这些检索方法可以看做是 XML 检索的先驱。13.4 节将介绍 XML 检索。需要指出的是数据库和信息检索研究人员都共同关注将 XML 作为文档标记标准所带来的一系列问题［608］。13.4 节讨论了信息检索研究界所关注的*面向内容的* XML 检

⊖ http://inex.is.informatik.uni-duisburg.de/。

索（content-oriented XML retrieval）问题。13.5 节概括地介绍了 INEX，它定义了 XML 检索效果的标准评价方法。13.6 节通过介绍为访问 XML 文档而开发的查询语言，对本章进行总结。

13.2 结构化能力

从 20 世纪 80 年代末一直持续到整个 20 世纪 90 年代，多种结构化文本检索模型出现在文献中。它们包含三个主要部分 [761]：1）文本的模型，定义了字符集、同义词、禁用词以及词干提取；2）结构的模型，定义了标记语言、索引结构、结构类型；3）查询语言，定义了什么可以询问以及结果是什么。这里我们不对这三个部分如何在不同的结构化文本检索模型中加以处理进行综述。而是关注它们的结构化能力，这与文本成分是如何定义的相关。这里的文本成分通常指的是连续的文本部分，涉及元素（element）、区域（region）、片段（fragment）或者段（segment）。我们对结构化文本检索的结构化能力在如下三个方面进行对比：显式和隐式结构、静态和动态结构，以及单一和多层次结构。

13.2.1 显式和隐式结构对比

绝大多数的结构化文本检索模型是基于文档的显式结构定义的。换句话说，这些模型的工作基础是文档已经结构化，这种情况下文档的章、节和标题等都被清晰地标记出来。 546
这些结构通常使用标记语言提供。因此，这些模型假设文本已经根据事先定义的模式（DTD 或者 XML 模式）进行了很好的表示。显式结构确保这些模型知道哪些区域是相互嵌套的，并且可以准确地访问结构关系，如直接祖先关系（在 XPath 中的父子关系，参见 13.6.3 节）。假如一个用户正在寻找包含“red wine”的节，就可以通过如下的查询进行表示：

```
section CONTAINS "red wine"
```

这个查询会返回所有包含“red wine”的节。

在隐式结构的情况下，文档的结构信息与文本内容没有明确区别。文档通过一些标签序列进行描述，单词标签和结构标签是不区分的。因此，结构元素是通过检索时查找结构标签的开始和结束来构建的。那些包含了特定的开始标签和对应的结束标签，并且满足内容要求的区域会被检索出来。一个涉及隐式结构的查询例子如下：

```
("<section>" FOLLOWING "</section>") CONTAINS "red wine"
```

section 元素只出现在查询阶段。需要指出的是，上面这个查询与指定了以一个特定单词开始，以另外一个单词结尾的区域查询没有区别。换句话说，对单词标签和结构标签的处理没有区别。

13.2.2 静态与动态结构对比

文本检索模型准许在查询中动态地定义结构，这样可以返回并没有被显式地在文本文档中进行标记的元素或者区域。在 XML 查询语言（XQuery）和 XML 全文查询语言（XQuery Full-Text）中（在 13.6.3 节进行描述），通过元素构造实现。但是，在其他模型中，动态结构是模型中自然的一部分（例如那些包含由类似 Word 的应用程序所控制的一些隐式结构的模型）。考虑下面这个结构化文档：

SPIRE, “Patagonia, Chile”, 2001, The conference ..... .

假设上面的这个文档可以利用如下语法作为文档模式 [648] 显式地进行结构化：

```
entry        :=  conference ', ' area ', ' year ', ' content'.'  ;
conference   :=  text ;
area         :=  '"' text '"' ;
year         :=  digit digit digit digit ;
content      :=  text ;
text         :=  ( letter | ' ' )+ ;
```

所有的文档实例必须遵照这个语法。已解析的字符串实例叫做“p-string”。有了上述模式，area 符号还是不能区别国家“Chile”和地区“Patagonia”。这种区分可以在查询时通过引入
547 一小段语法片段 **AreaG** 来解决，它可以从 area 字符串中分析出地区和国家：

```
AreaG  :=  { area     :=  ( region ' , ' )+ country ;
             country  :=  letter + ;
             region   :=  letter + ; }
```

p-string 模型提供了一种用于增加额外语法片段的简单查询语言。给定文档 d_j，下面所列的查询会返回一个 p-string，它包含给定国家和地区的 area 元素：

(area *in* d_j) *reparsed by* AreaG

这种结构可以用于搜索所有召开过 SPIRE 会议的智利地区。p-string 模型利用正则表达式匹配作为核心语言原语，它可以比 XQuery 和全文 XQuery 更容易地指定动态结构。

13.2.3 单一层次结构与多层次结构对比

大部分的结构化文本检索模型中使用的结构类型是分层组织方式。简单来看，使用隐式结构的文本检索模型可以认为是一种单一层次结构。当使用多层次结构时，维护隐式的多层次结构要复杂得多。

基于显式结构的方法假设多层次结构出现在同一个文档中 [299, 648, 1179]。每种层次可能为不同的目的服务。例如一个层次可能表示文本的逻辑层次（章、节、小节），而第二个层次可能表示布局结构（栏、页）。在单一层次中，结构元素要么不相交要么内部相互嵌套，但是不同层次的元素之间可能部分重叠，例如一个小节可能从某一页的中间开始到下一页的结尾结束。层次结构通常被称为“独立标注（stand-off annotation）”，用来强调结构信息（或标记语言）与文本内容是分别建模的。

然而，通过这些模型进行查询只可能在某一单一层次结构中进行。由于有很多不确定情况的出现，不可能对混合层次结构进行查询。也就是说，查询必须在一个层次结构上定义，然后映射到其他层次结构上。一个典型的例子是定位书中符合某个条件的结构所出现的实际页码。

这种情况已经随着 Alink 等人的工作而改变了 [27]，其中引入了基于 XPath 的导航步骤，它用来进行一个层次结构到另外一个层次结构的移动。例如，查询

```
$doc//paragraph[
    ./select-narrow::Verb CONTAINS "hiking" and
    ./select-narrow::Region CONTAINS "Patagonia"
]
```

从 paragraph 元素开始导航到另外一个包含 verb 元素“hiking”的层次结构，再到包含
548 region元素“Patagonia”的层次结构。

13.3 早期文本检索模型

现在我们来讨论两种早期的结构化文本检索模型，一种是基于非覆盖列表（non-overlapping lists）的模型，另外一种是基于相邻结点（proximal nodes）的模型。这两种模型对

结构化文本检索中的主要问题和需要权衡的因素提供了很好的概述。

我们使用项匹配点来表示文本中符合用户查询的词的位置。因此，对于用户输入的简单查询“red wine”，如果在文档 d_j 中有三个位置出现了，那么我们就说 d_j 包含三个项匹配点。而且，我们使用术语“区域”（region）来表示一段连续的文本，术语“结点”（node）表示文档的结构成分，如章、节和小节等。因此，结点是文档的作者和想要检索文档集的用户都知道的、具有预定义拓扑属性的区域。

13.3.1 基于非覆盖列表的模型

Burkowski[298，299] 将每篇文档的内容分割成多个非覆盖的文本区域，并收集在一个列表中。由于存在多种方式将文本切分为非覆盖的区域，因此会产生多个列表。例如，可以有由文档中所有章组成的第一个列表，由文档中所有节组成的第二个列表，以及由文档中的所有小节组成的第三个列表。这些列表分别保存在不同的数据结构中。虽然在同一个列表中的文本区域是没有覆盖的，但是不同列表中的文本区域可能有覆盖。图 13-1 给出了对于同一个文档的四个独立列表的例子。

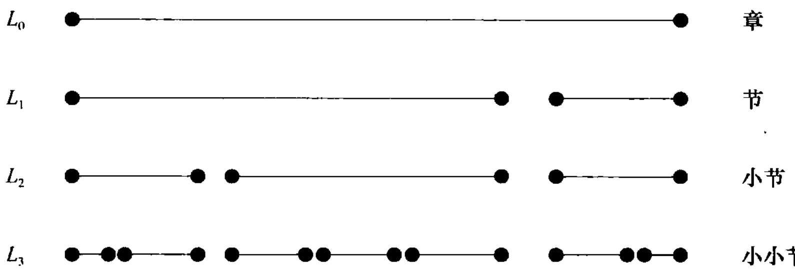


图 13-1 使用四个单独（扁平）的索引列表来表示文档中的文本结构

为了查找索引项和文本区域，构造单一倒排索引（inverted index）（参见 9.2 节），其中的每个结构成分都作为索引中的一个条目。一个文本区域列表作为记录列表与每个条目关联。此外，对于文本中的单词，这个列表也可以很容易地与传统的倒排索引合并。因为文本区域是非覆盖的，所以可以支持的查询种类就很简单：1）选择一个区域包含某个词（并且不包含其他区域）；2）选择一个区域 A 不包含任何其他区域 B（B 所属的列表与 A 的不同）；3）选择一个不包含在其他区域里的区域。 549

13.3.2 基于相邻结点的模型

Baeza-Yates 和 Navarro[1178，1179] 提出了可以在同一个文档中定义独立层次（非扁平）索引结构的模型。这些索引结构的每一个都是由被称为结点（node）的章、节、段落、页、行组成的严格层次结构。每一个结点都有对应的文本区域。此外，两个不同的层次结构可能会指向重叠的文本区域。

给定一个涉及不同层次结构的用户查询，编辑后的结果都是由包含在一个层次结构中的结点组成的。因此，结果不能由来自于两个不同的层次结构的结点组成。这样是为了加快查询处理速度，而所付出的代价是较低的表达能力。但是需要注意的是，对于层次结构，在答案集合中允许出现嵌套的文本区域（来源于同一个层次结构）。

图 13-2 给出了一个层次化索引结构的例子，它包含四个层次（同一个文档中的章、节、小节、小小节）和关于单词“Patagonia”的倒排表。倒排表中的条目表示了单词“Patago-

nia”在文档中所出现的所有位置。在层次结构中，每个结点包含了其相关的结构成分（章、节、小节或小小节）在文本出现的位置。

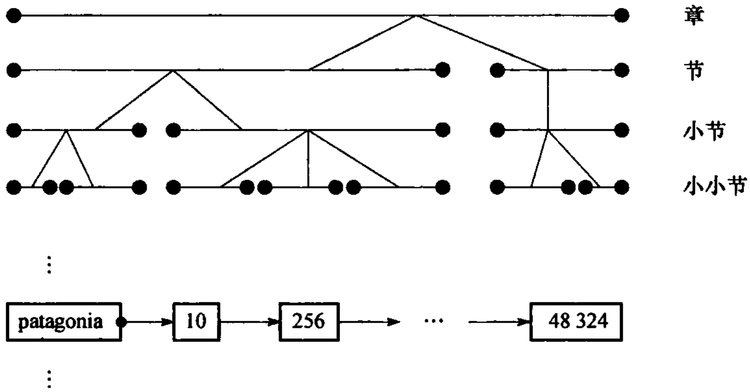


图 13-2 表示结构成分的层次索引和表示单词的扁平索引

查询语言可以使用正则表达式来检索字符串，通过名字来查询结构成分（例如检索章），还可以将它们进行组合。从这个意义上说，该模型可以看成是表达能力和效率之间的平衡。查询语言的表达能力在一定程度上受到限制，但却可以加快查询处理，其途径是首先搜索符合查询的字符串成分的那些部分，随后再对这些部分是否符合查询中的结构成分进行评估。考虑如下查询：

```
(*section) with ("Patagonia")
```

检索包含了单词“Patagonia”的所有节、小节或者小小节。简单的查询处理策略就是遍历查询项“Patagonia”的倒排表，对列表中每个条目（表示查询项“Patagonia”出现在文本
550 中的位置），搜索层次索引，查找包含这个查询的节、小节和小小节。

一个更复杂的查询处理策略如下。对于在“Patagonia”列表中的第一个条目按照原来的方法搜索层次索引。这就意味着向下遍历层次结构，直到没有更多成功的匹配出现（或达到层次的底部）。将最后匹配的结构成分作为最内部的匹配单元（innermost matching component）。在第一次搜索完成后，对于倒排表中接下来的条目并不是从头开始。而是，首先检查最内部的匹配单元是否也符合当前条目。如果是符合的，我们就可以立即断定它之上（在层次结构上）的结构成分也符合要求。接下来，对所有条目依次进行上面的处理。需要注意的是查询处理之所以可以加快，是因为相邻结点在一次检索中只会被查找一次。这也是为什么叫做“相邻结点”的原因。

在基于相邻结点的模型中可以构造比基于非覆盖列表的模型更加复杂的查询。然而，为了加快查询处理，只查看相邻的结点，而同时也限定了查询结果集（所有的结点必须来源于同一个层次结构）。

13.3.3 结构化文本结果排序

绝大多数早期结构化文本检索模型的检索结果并没有排序。直到 2002 年，随着开创了 XML 检索评价的 INEX 的提出（参见 13.5 节），这个局面才得到改变。面向内容（相对于面向数据）的 XML 检索方法现在都包含由查询指定的、基于内容和结构成分的元素排序，将在 13.4.3 节中介绍。

13.4　XML 检索

由于 XML 标记语言的成功，如今 XML 检索几乎是结构化文本检索的代名词，虽然 XML 检索只解决了显式的单一层次结构问题。XML 检索是结构化文本检索的一种特例，在其中文档的结构标记使用 XML。虽然 XML 检索方法自 20 世纪 90 年代末以来有了一定的发展，但是主要贡献都是随着 INEX 的建立而取得的，它提供了测试集，为评价和比较 XML 检索方法而设立了专门的论坛。本节将概要性地介绍在 INEX 竞赛下开发和评价的 XML 检索方法。

13.4.1　XML 检索中的挑战

在 INEX 中，XML 检索任务定义为：

XML 检索任务：*利用 XML 文档的层次结构来确定最佳的文档成分，即根据用户查询返回最佳的 XML 元素作为结果。此外，这些结果应当根据它们与查询的相关程度进行排序。* 551

我们讨论在检索这些最佳元素时所要面临的主要挑战。

正如我们在第 3 章中所讨论的，经典的信息检索模型使用索引项的统计信息，如文档内项频（Term Frequency，TF）和反比文档频率（Inverse Document Frequency，IDF）来对文档排序。TF 表示的是索引项在一篇文档中出现的次数，用来反映这个索引项符合文档主题的程度，IDF 表示的是出现索引项的文档个数的倒数，用来反映它在不同文档之间的区分度。

XML 检索所使用的索引算法也需要类似的索引项统计信息，但却是在元素级。我们可以简单地通过将文档替换为元素（element）然后计算出所谓的*元素内项频*（within-Element Term Frequency，ETF）和反比元素频率（Inverse Element Frequency，IEF）。然而，由于 XML 文档的嵌套性质，这就带来一个问题。例如，假设一个节元素中包含两个段落元素。索引项出现在一个段落中就意味它也出现在这一节中，这些因素在计算索引项的 IEF 值时必须要考虑进去。为了解决这个问题，提出了很多用来估计元素频率的索引策略，我们将在 13.4.2 节中进行讨论。

并不是所有的元素类型在作为查询结果时都满足用户需求。有些元素可能不适合出现在检索结果中，或者因为太小或者因为这些元素中不包含有信息的文本。这就要求在对 XML 元素进行排序时，要考虑它们的大小和类型。哪些类型的元素作为检索结果最好，以及如何使用这些信息来对元素进行排序都是具有挑战的问题。例如，将元素的长度在排序时考虑进去（参见 13.4.3 节）已经证明对提高 XML 检索结果很重要。使用选择性索引（selective indexing）（参见 13.4.2 节）和在排序函数中使用参数（参见 13.4.3 节）都意味着在 XML 检索中指定特定的元素类型。

XML 文档不仅是包含了不同类型和不同大小元素的文档，通过 XML 标记语言的逻辑结构也反映了元素之间的关系。一个非根结点的元素会有一个父元素，父元素本身也可能还有父元素。类似的，非叶子结点的元素包含子元素等。这些元素之间的关系也可以用来提高 XML 检索。例如，在科学文献文档集中，我们可以做这样的假设：一篇文章的“摘要”部分比“未来工作”部分更能代表这篇文章的内容。在上下文方法中（参见 13.4.3 节），我们可以看到将根元素与其子元素之间的关系纳入到元素排序中，可以提高检索性能，且独立于用来估计相关性的检索模型。

对一个元素与查询的符合程度进行评分的任务，与从多个重叠的相关元素中抽取出作为答案的最佳元素的任务是不同的。这是由于 XML 文档天然的嵌套结构。如果一个元素与查询相关，那么它的父元素也会被认为与查询相关（可能在不同程度上），因为这两个元素有重叠的文本。然而，应当避免在返回一个段落的同时，还返回它所处的节，导致让用户多次得到相同的信息。决定返回哪些元素给用户是由应用和用户模型决定的。例如，在 INEX 中，最好的元素不仅是最相关的，而且还是符合主题要求的（即没有讨论不相关主题）。在
552 13.4.4 节中我们将会讨论去除重叠的方法。

最后一个挑战是如何解释结构约束。XML 检索早期工作要求检索结果必须完全符合查询限制。但是在查询中指定结构约束不是件容易的事情，因为在实际中，XML 文档集包含大量标签名。用户不太可能对要检索文档集的结构有非常清晰的认识，如果有的话往往这些方面并不是他所关心的内容。因此，在 INEX 竞赛中将结构约束看做是结构提示，即看做在哪里（哪种元素）可以找到相关内容。如何在对元素排序的时候使用结构提示将在 13.4.3 节中进行讨论。

13.4.2 索引策略

与“扁平的”文档检索不同，在 XML 检索中，没有先验的（a priori）的固定检索单元。整篇文档、其中的一个部分（如其中某一章），或者一个部分的部分（如某一节中的一个段落）都可以构成一个查询的潜在答案。最简单的、提供检索所有颗粒度元素的方法是，对所有元素都建立索引。因此，一个元素对应于一个文档，传统的信息检索索引技术都可以使用。接下来，每个元素的索引项统计信息（ETF 和 IEF）都可以根据连接元素及其后代的文本计算出来。

关于反比元素频率（IEF）的计算，上述方法忽略了嵌套元素的问题，即一个索引项的 IEF 值要考虑所有包含这个索引项的元素以及这些元素的祖先（如 [1476]）。或者，IEF 可以跨越同种元素（如 [1579]）或跨越多个文档（如 [389]）进行估计。前面一种方法大大减少了嵌套元素对 IEF 值计算的影响，但并不能消除同一类型的元素可以相互嵌套的影响。后面这种方法与使用反比文档频率相同，完全避免了嵌套元素。在 [1331] 中报告的实验结果表明在语言模型框架下，相比基于元素的 IEF 估计，跨越文档方式估计的 IEF 的结果有微小提高。利用 BM25 概率排序改造的 XML 检索的实验结果 [281] 则表明通过全部元素[⊖]估计得到 IEF 比仅通过同类元素估计得到的 IEF 结果要好。在目前阶段，尚不清楚什么是最好的 IEF 估计方法，仍需要进一步研究。

我们并非利用元素中的连续文本来计算统计信息，而是聚合元素自身的文本以及其子元素的统计信息来计算索引项的统计信息（如 [659，1286]）。而这样可以避免 IEF 在嵌套元素中的计算问题。此外，在聚合中可以引入另外的参数，如元素间的关系或者元素类型等。在 13.4.3 节中将讨论基于聚合的排序，即利用元素的聚合表示来对元
553 素排序。

对所有元素进行索引会导致索引很大，且包含很多冗余信息。如在 INEX 2002—2004 文档集中包含了大约 12 000 篇文章，共计 800 万个元素，其中很多嵌套在一起。另外一种方法是仅索引叶子元素。这意味着，仅对叶子元素计算索引项统计信息，接下来可以对叶子元素进行排序。对于非叶子元素的排序需要传播机制（propagation mechanisms）

⊖ 由于速度原因，并没有考虑所有的元素，但是大部分元素都考虑在里面。

（参见 13.4.3 节）将它们子元素的得分汇总起来［620］。这也避免了跨越嵌套元素计算 IEF 值的问题，使得索引更容易管理。但是，对于非叶子结点需要有效并且高效的传播机制。

通常我们会丢弃掉小于某个给定阈值（一般用单词数量表示）的元素，认为它们不是有意义的检索单元［1476］。虽然所谓的小元素不会被返回，但是在实际采用传播算法来计算非叶子元素得分的方法时，它们仍会影响那些包含它们元素的得分，所以需要对其进行索引［1427］。一个称为选择性索引（selective indexing）的相关策略［389，1098］，只对那些在过去的相关数据中具有较高相关性的元素类型进行索引。

选择性索引策略［1098］可以与 13.3.1 节中的非覆盖列表模型进行对比。在它们的策略中，对每个选择出来的元素类型都单独构建索引。例如，对于一个科学文献文档集，这些类型就可能包括正文、摘要、节、小节和段。对每一个索引的统计信息也分别计算。因为每个索引条目指向同一类型的元素，它们的索引项统计信息可能比包含所有类型的索引更均匀。除了可以提供更一致的索引项统计外，这种方法还在很大程度上降低了嵌套结构带来的索引项统计问题。在检索阶段，查询可以并行运行在每一个索引上，返回的结果列表（每个索引一个）最后会合并到一个结果列表中。我们将在 13.4.3 节中进行讨论。

目前尚不清楚哪一种索引策略是最好的，因为很明显这些方法取决于文档集、元素的类型（即 DTD）以及它们之间的关系。此外，对于索引策略的选择也会影响到排序策略。在一个统一的、可控的环境下，研究所有索引策略来判断哪些排序策略有最佳表现，将会是一个吸引人的研究。

13.4.3　排序策略

第 3 章介绍的检索模型大多都已经经过改造来适应 XML 检索。这些模型仅基于元素内容来估计元素的相关程度。在实际问题中，特别对于长文档，利用从上下文元素（如父元素）中得到证据也可以提高检索性能。同时根据索引策略，需要传播、汇聚以及合并等特定策略对各种颗粒度的元素进行排序。最后，对于内容-结构查询，必须处理结构约束，使得结果既能满足查询的内容要求也能满足查询的结构准则。 554

```
<article>
<sec>
 <subsec>
   <p> ... wine ... patagonia ... </p>
   <p> ... wine ... </p>
   <p> ... patagonia ...  </p>
 </subsec>
 <subsec>
   <p> ... </p>
   <p> ... </p>
 </subsec>
</sec>
<sec>
 <p> ... </p>
 <p> ... wine ...  </p>
 <p> ... </p>
 <p> ... </p>
</sec>
</article>
```

图 13-3　XML 文档样例

在本节中，我们通篇使用图 13-3 中的 XML 文档样例来进行说明。这个 XML 文档表示了一个由两节组成的文章。其中第 1 节包含两个小节，第 1 个小节中包含了三个段落，第 2 个小节中包含了两个段落。第 2 节由四个段落组成。图 13-4给出了这个样例文档的树结构，为了能够容易地区分这些元素，它们都给定义了唯一的标识。

1. 元素评分

所有的排序策略都需要利用评分函数对元素与查询间的相关度进行评分。利用传播机制（本节的后半部分进行讨论）的评分函数只应用于叶子结点；其他情况下，评分函数应用于所有可以检索到的元素。评分函数通常基于标准的信息检索模型，例如，向量空

间模型、BM25 以及语言模型，它们经过改造引入了 XML 特有的特征。例如，这里我们介绍如何将 XML 特有的特征与语言模型结合。这个方法受到 Sigurbjörnsson 等人工作的
555 启发 [1476]。

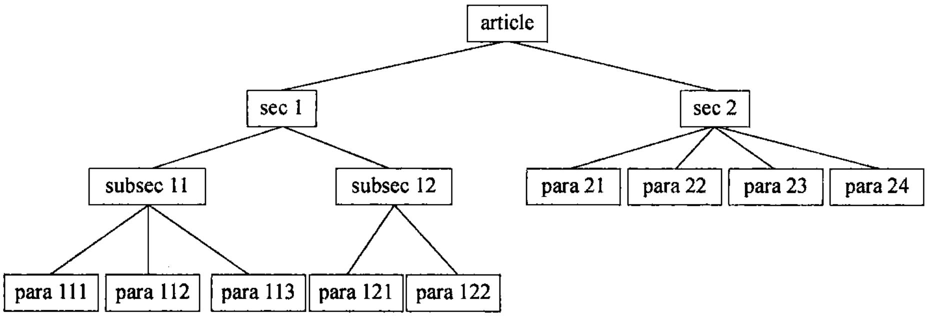


图 13-4 XML 样例文档的树形结构

给定一个查询 $q=(k_1, k_2, \cdots, k_n)$，包含 n 个索引项 k_i，一个元素 e，以及其对应的元素语言模型 M_e，元素根据 $P(e|q)$ 按照降序的方式进行排序：

$$P(e|q) \propto P(e)P(q|M_e)$$

其中 $P(e)$ 是关于元素 e 相关性的先验概率，$P(q|M_e)$ 是根据元素语言模型 M_e 生成查询的概率。利用多项语言模型 (multinomial language model) (参见 3.5.2 节)，$P(q|M_e)$ 可以按照如下方式计算：

$$P(q|M_e) = \prod_{k_i \in q} P(k_i|M_e,\lambda)$$

如果索引项的概率计算使用 3.5.2 节中介绍的 Jelinek-Mercer 平滑方法，可以得到：

$$P(k_i|M_e,\lambda) = \lambda P(k_i|e) + (1-\lambda)P(k_i|C) \qquad (13\text{-}1)$$

其中 $P(k_i|e)$ 是查询项 k_i 在元素 e 中的概率，$P(k_i|C)$ 是查询项 k_i 在整个文档集中的概率，λ 是平滑参数。$P(k_i|e)$ 是根据元素项频 ETF 得到的元素模型，$P(k_i|C)$ 是根据反比元素频率 IEF 得到的文档集模型。如何计算这些概率可以参见 3.5.2 节。

假设查询包含两个索引项 q=(wine，patagonia)。表 13-1 中列出了根据样例 XML 文档计算得到的查询项 *wine* 和 *patagonia* 的 $P(k_i|e)$ 值。内部元素的索引项统计信息是根据元素自己的内容及其后代元素计算得到的。*length*(e) 表示元素 e 的长度，即 e 中包含多少个索引项。一个内部元素的长度是其子元素长度的总和。假设 $\lambda=0.8$，表 13-1 中列出了相应的索引项 *wine* 和 *patagonia* 经过平滑的 $P(k_i|M_e)$ 值 (之后的计算需要这些值)。它们通过式 (13-1) 计算得到，其中 $P(k_i|C)$ 的值由表 13-1 中给出。这里我们省略掉了不包含查询项的元素。这个工作也可以通过非平滑的概率完成，于是在这些元素 e 中 $P(q|M_e)=0$。图 13-5a 中给出了样例文档中的元素排序列表。

表 13-1 XML 样例文档的 $P(k_i|e)$、$P(k_i|C)$、$P(k_i|M_e)$ 和 *length*(e) 统计信息

元素	$P(wine\|e)$	$P(wine\|M_e)$	$P(patag...\|e)$	$P(patag...\|M_e)$	*length*(e)
para111	0.2	0.180	0.333	0.327	15
para112	0.6	0.500	0	0.060	10
para113	0	0.020	0.25	0.260	12
para121	0	0	0	0	8
para122	0	0	0	0	10

（续）

元素	$P(\text{wine}\mid e)$	$P(\text{wine}\mid M_e)$	$P(\text{patag}...\mid e)$	$P(\text{patag}...\mid M_e)$	$length(e)$
para21	0	0	0	0	20
para22	0.5	0.420	0	0.060	14
para23	0	0	0	0	10
para24	0	0	0	0	18
subsec11	0.243	0.215	0.216	0.233	37
subsec12	0	0	0	0	18
sec1	0.155	0.144	0.138	0.170	58
sec2	0.113	0.110	0	0.060	62
article	0.127	0.122	0.063	0.111	126
$P(k_i\mid C)$	0.1	$\lambda=0.8$	0.3	$\lambda=0.8$	

这里，我们假设了一个恒定的先验相关概率，因此 $P(e)$ 可以省略，也就是 $P(e\mid q)\propto P(q\mid M_e)$。然而，在上面的语言模型框架下，XML 特有的特征可以通过 $P(e)$ 引入。例如，考虑 XML 文档中元素大小的范围（例如从段落到文章），特别是在 XML 文档集中，小元素的分布偏置会比较严重（如段落的数量比节的数量多很多），可以通过如下方式引入对于长元素的偏置 [866]： 556

$$P(e)=\frac{length(e)}{\sum_{e'} length(e')}$$

如果我们假设 $\sum_{e'} length(e')=400$，那么 $P(\text{para11}\mid q)\propto 15/400\times 0.0588=0.0022$。组成样例 XML 文档的元素按照图 13-5b 进行排序，其中我们可以看到，较大的元素的排序会靠前。

其他可以引入语言模型框架的 XML 特有特征包括在一个元素中话题切换的数量 [74]，以及元素的路径长度 [785] 等。总之，在 XML 检索中，元素长度是重要的因素，其他 XML 特有的特征在不同的检索任务中可能也有一定的帮助 [74]。

a)	b)	c)
0.0588 para11	0.0046 subsec11	0.0361 para11
0.0500 subsec11	0.0042 article	0.0317 subsec11
0.0300 para112	0.0036 sec1	0.0217 para112
0.0252 para22	0.0022 para111	0.0193 para22
0.0246 sec1	0.0010 sec2	0.0190 sec1
0.0135 article	0.0009 para22	0.0135 article
0.0066 sec2	0.0008 para112	0.0100 sec2
0.0052 para113	0.0002 para113	0.0093 para113

图 13-5　不同的排序方式：a）仅使用元素评分；b）元素评分结合长度偏置；c）上下文研究法

虽然元素特有的特征在估计查询与元素的相关性之间是有帮助的，但是结果的评分通常情况下并不直接用于产生对查询元素的排序列表。接下来，我们会讨论用来产生最后的元素排序列表的附加策略。 557

2. 上下文研究法

出现在某些类型元素中的索引项通常构成另外一些类型元素索引项的子集。例如，在 INEX 2002—2004 文档集中，段落索引中的索引项比文章索引中的索引项少 25% [1098]。将索引项词汇表的这种变化考虑进去，会对 XML 检索有一定的帮助，特别是按照文档层次结构向下估计元素相关性的时候（趋势变小）。可以通过考虑元素的上下文、它的父结点、它的某些或者全部祖先结点，或者整篇文档（根结点）来解决。

元素的上下文可以提供很多元素是否关于什么内容的有用证据。这是因为文档中的所有索引项，不论是否在这个元素中，都可以用来对这个元素是否与给定的查询相关评分。比如

一个元素中不包含所有的查询项，但是它所在的文档包含全部的查询项，这个事实对于判断相关性十分重要。这个策略可以通过结合元素得分与文档得分来实现。将元素得分与其上下文得分进行结合的方法叫做上下文研究法。

最常见的上下文方法是用包含这个元素的文档（根结点）作为上下文。这就意味着将通过 XML 检索模型得到该元素的得分与包含该元素的文档的得分进行综合。最简单的综合方法就是取它们的平均值 [73]。比率因子也可以引入进来，这样可以强调一个得分比另外一个重要 [1098]。

我们将上下文方法用于我们的例子中。图 13-5a 给出了只使用元素自身的评分策略所估计的得分。如果我们假设元素和文档（在我们的例子中就是文章）的得分是一样重要，那么最后排序结果就如图 13-5c 所示。例如，parall1 的新得分就变成了 (0.0588+0.135)/2=0.0361。需要注意的是，图 13-5c 与图 13-5a 的排序是一样的，因为在这里我们只有一个文档，但是不管怎样我们都能看到最初的方法和上下文方法之间的区别。

一个元素的部分或者全部祖先都可以用于上下文。例如父元素可以单独用于上下文。所有的祖先元素一起也可以用于上下文 [73]。将多个上下文合并起来需要面对的一个问题是需要设置不同上下文的权重比例。机器学习方法可以用于这个目标 [1642]。

在 XML 检索中，使用元素的上下文来估计它的相关性需要很好地获得元素间的关系。特别是在长文档中，如果可以很好地利用这些关系可以稳定地提高检索性能。

3. 传播

在仅对叶子元素进行索引的机制中（参见 13.4.2 节）需要用到传播机制。首先，叶子元素的得分通过索引直接求出。非叶子元素的得分（内部元素，包含根元素在内）是通过传播机制估计得到的。内部元素得分是基于它的后代元素的得分计算而来的。传播从叶子元素

558 开始，按照文档树结构向上传播。

假设 e 是一个非叶子（内部）元素，e_l 是 e 所包含的叶子元素[⊖]，q 表示查询，$score(\cdot)$ 表示用来根据元素相关度估计对元素排序的得分函数。对于 e_l，$score(e_l, q)$ 可以通过 e_l 的索引直接估计得到（例如使用向量空间模型），而 $score(e, q)$ 则需要通过传播机制计算得到。最常见的传播机制是通过对得分加权求和得到，这其中的变化就是对权重的定义和估计。

元素包含的子结点的多少可以用来作为权重，例如，在 GPX 方法中 [620]，$score(e, q)$ 通过下式计算得到[⊜]：

$$score(e,q) = D(m) \times \sum_{e_c} score(e_c,q) \tag{13-2}$$

其中 e_c 是 e 的子元素，m 是元素 e 中检出的子元素数量（所有 e_c 满足 $score(e_c, q)>0$）。如果 $m=1$(e 只有一个检出的子元素)，$D(m)=0.49$；其他情况下，$D(m)=0.99$。$D(m)$ 的值叫做衰减因子（decay factor），它取决于检出的子元素的数量。如果 e 只检出一个子元素，那么衰减因子的值为 0.49，表示它的得分应该比子元素低。如果 e 检出多个子元素，那么衰减因子的值为 0.99，表示它通常应该比其子元素的得分高。因此，如果一节中只有一个相关段落，那么它的相关性比这个段落本身的相关性要小（简单地只返回这个段落比返回没有包含其他信息的整节要好）；如果一节中包含了多个相关段，那么它应该比任何一个段落的排序都靠前（可以准许用户从返回的节中访问这些段落）。

⊖ e 和 e_l 之间存在路径，其中 e_l 是这个路径的最后一个元素，也就是说 e_l 没有子元素。

⊜ 当权重用子元素的数量来反映时，所有子元素的权重是相同的，因此可以放在求和外。

我们使用 GPX 技术来举例说明传播机制在我们的 XML 样例文档上的应用。我们使用如图 13-5a 所示的、仅使用元素评分机制[㊀]的叶子结点得分。subsec11 包含三个子元素，因为 $m=3$，所以 $D(3)=0.99$，因此 $score$(subsec11，{wine，patagonia})$=0.99\times(0.0588+0.0300+0.0052)=0.0931$。排序结果如图 13-6a 所示。GPX 传播算法使得 subsec11 的得分现在要高于它的三个段落，这是有意义的，因为这些段落可以通过 subsec11 访问到。我们还可以看到 sec2 依然比 para22 的排序低。

a)	b)
0.0931 subsec11 (0.99)	0.0588 para111
0.0588 para111	0.0503 subsec11
0.0574 article (0.99)	0.0300 para112
0.0456 sec1 (0.49)	0.0252 para22
0.0300 para112	0.0126 sec1
0.0252 para22	0.0068 article
0.0123 sec2 (0.49)	0.0052 para113
0.0052 para113	0.0016 sec2

图 13-6　a) 使用 GPX 传播机制进行排序，衰减因子 $D(m)$ 在括号中列出；b) 使用聚合策略排序结果

距离定义为从内部结点到叶子结点间路径的长度，也可以用做权重。在 XFIRM 系统 [1427] 中，距离与上下文策略一起使用在传播机制中。对于给定的查询项 q，内部元素 e 的简化评分方式如下：

$$score(e,q)=\rho\times m\times\sum_{e_l}\alpha^{d(e,e_l)-1}\times score(e_l,q)+(1-\rho)\times score(root,q) \tag{13-3}$$

其中 m 是 e 包含的检出的叶子元素的数量（换句话说，这些叶子元素 e_l 满足 $score(e_l,q)>0$），$d(e,e_l)$ 表示 e 和 e_l 在文档树上的距离，$score(root,q)$ 表示 $root$ 元素关于查询 q 的得分，ρ 用来控制在上下文策略中元素得分相对于文档得分的重要程度。

利用传播机制可以得到很好的检索性能，特别是 GPX 系统所实现的。虽然很简单，但是 GPX 传播机制在 INEX 竞赛中的检索任务中得到了最好的性能，体现了它对于 XML 检索的通用性。

4. 聚合

聚合是基于 Chiaramella 等人在结构化文档检索上的工作 [371]。其基本思想是 XML 元素（结构成分）的表示可以看做其自身（如果存在的话）以及结构相关元素（如果存在的话）的内容表示的聚合。XML 检索中常见的做法是将元素自身的内容与其子元素的内容聚合起来。检索将基于这些聚合起来的表示。

元素自身内容表示是使用标准的索引技术生成的，其中聚合函数用来生成非叶子元素的表示。聚合函数可以包含某些参数（被称做加强因子（augmentation factor）[659] 或者可及度权重（accessibility weight）[1379]）来确定这个元素的表示如何被子元素所影响（对其贡献的测度，例如一节对其所属章的影响）。

为了说明 XML 检索中的聚合机制，我们介绍一种基于（简化）语言模型框架的方法，该方法受到 Ogilvie 和 Callan 工作的启发 [1286]。在这个框架中，每个元素都由一个语言模型来建模。对于元素 e，查询项 k_i 的概率根据这个元素自身内容的语言模型 $M_{e_{own}}$ 得到：

$$P(k_i|M_{e_{own}})=(1-\lambda)P(k_i|e_{own})+\lambda P(k_i|C) \tag{13-4}$$ 559～560

其中 λ 是平滑参数，用来控制背景文档集模型 C 的影响。

在我们的例子中，只有叶子元素包含自身内容，所有内部元素的内容是由它的子元素得到的。如果我们假设表 13-1 中给出了叶子元素的索引项统计信息，并且令 $\lambda=0.8$，那么 $P(k_i|M_{e_{own}})$ 的值如表 13-2 左半部分所示[㊁]。

㊀ GPX 算法对叶子元素使用了不同的评分函数。

㊁ 这些值与表 13-1 中 $P(k_i|M_e)$ 的值相同。对于不包含查询项的元素我们都有同样的假设，比如对于这类元素设置 $P(k_i|M_{e_{own}})=0$。

表 13-2 对叶子元素的索引项统计信息 $P(k_i|M_{own})$ 和通过聚合策略得到的非叶子元素的索引项统计信息 $P(k_i|M_e)$。注意：对于叶子元素 $P(k_i|M_{own})=P(k_i|M_e)$

	元素	$P(\text{wine}\mid M_{own})$	$P(\text{patagonia}\mid M_{own})$		元素	$P(\text{wine}\mid M_{own})$	$P(\text{patagonia}\mid M_{own})$
叶子元素	para111	0.180	0.327	内部元素	subsec11	0.233	0.216
	para112	0.500	0.060		sec1	0.117	0.108
	para113	0.020	0.260		sec2	0.105	0
	para22	0.420	0.060		article	0.111	0.054

现在我们假设元素 e 包含多个子元素 e_j，每个子元素都包含自身的语言模型 M_{e_j}，那么聚合函数可以通过对这些语言模型的线性插值方法得到：

$$P(k_i|M_e) = \sum_{e_j}\omega_j P(k_i|M_{e_j})$$

其中，$\sum_{e_j}\omega_j = 1$。参数 ω 用来对每个语言模型（即子元素）在聚合中的贡献建模。

对于给出的样例，我们假设每个子元素的贡献是相同的，意味着将 $P(k_i|M_{e_j})$ 除以子元素的数目。表 13-2 中给出了聚合后的索引项统计信息。例如，因为 subsec11 包含 3 个子元素，所以对于索引项 wine 聚合后的结果为 $P(\text{wine}|M_{subsec11}) = \frac{1}{3}\times(0.180+0.500+0.020)=0.233$，其中 $\omega=1/3$，因为我们假定所有包含进来的元素的语言模型的贡献是相同的。

元素的排序按照每个元素生成查询的概率来产生。我们可以使用与产生图 13-5a 的排序相同的公式，它仅仅是把查询中的每个索引项的概率 $P(k_i|M_e)$ 连乘起来。图 13-6b 中给出
561 了排序结果。

其他处理聚合的方法包含带域 BM25 模型 [1057] 和概率模型 [659，889]。早期方法（没有在 INEX 框架下评测）包含 [967] 和 [1165]。聚合方法中一个重要的问题就是对于参数的估计问题（例如，对于上例中的 ω 值的估计）。

5. 合并

某些为 XML 检索开发的方法，对于给定的查询，会产生多个不同的排序列表，之后再将它们合并成一个排序表，并返回给用户 [177，980，1098]。

[1098] 介绍的方法使用选择性索引策略（参见 13.4.2 节），对于每种类型的元素单独建立索引（例如，对于一组科研论文，这些元素类型包括正文、摘要、节和段落）。我们假设检索模型在每个索引中对元素进行排序。对于每个索引会产生不同的排序列表。为了将这些列表合并起来，需要在归一化时考虑不同索引中元素大小的变化（例如，段落索引和正文索引）。为了这个目的，每个索引求出 $score(q, q)$，它表示当一个与查询相同的元素出现在文档集中时查询的得分。对于每个索引，元素的得分会根据 $score(q, q)$ 进行归一化，于是与查询完全相同的元素的得分就是满分 1。这就保证了根据不同的索引所得到的分数可以比较，也就使得元素可以通过归一化分数进行合并。

假设对于样例文档的每个索引（段落、节、子节和篇章），利用检索算法得到的（虚构）分数如表 13-3 所示。同样对于每个索引我们给出了 $score(q, q)$（虚构）得分，如果我们使用

$$\max\left(\frac{score(e,q)}{score(q,q)},1\right)$$

来对得分进行归一化，所得到的新得分如表 13-3 中括号中的部分所示。

表 13-3　对于每个索引的元素得分，归一化后的得分在括号中

段落索引	子节索引
para11 0.4 (0.5) para12 0.3 (0.375) para 113 0.2 (0.25) para22 0.5 (0.625)	subsec11 0.6 (0.857)
score(*q*, *q*)=0.8	*score*(*q*, *q*)=0.7
节索引	**篇章索引**
sec1 0.4 (0.513) sec2 0.2 (0.256)	article 0.3 (0.448)
score(*q*, *q*)=0.78	*score*(*q*, *q*)=0.67

562

6. 处理结构约束

在 INEX 中，结构约束可以看做是从哪里可以找到有用信息的提示。这种观点的原因包含两个方面。首先，大家都知道信息检索系统的用户不是总能，或者说大多数情况下都不能很好地表达他们的信息需求。这样的困难同样也存在于信息需求的结构准则上。例如，用户想找一个包含给定主题的段落，但是他可能没有意识到关于这个主题的有用信息分散于多个段落中，但是它们都属于同一节。这种情况下，返回这一节内容比单独的段落要有用得多。

其次，XML 信息检索研究界普遍坚信，在通常情况下，满足内容需求比满足结构要求更重要。比如，如果用户希望找到包含特定主题的一节内容，但是返回的是最满足内容需求的摘要部分，对于用户来说仍然是可以接受的。在 XML 检索中，某些针对处理结构约束的方法也都遵循这个观点。

第一种方法是构建一个标签同义词典。这个字典可以是基于句法的。例如，如果<**p**>对应段落类型，<**p1**>对应一组段落中的第一个，那么将<**p**>和<**p1**>看做等价的标签是符合逻辑的 [1097，1426]。词典也通过处理过去的相关性数据获得 [1127]。比如，如果在这样的数据集中，一个查询限定查找<**section**>类的元素，那么所有标记为与查询相关的元素类型，都可以考虑与<**section**>标签等价。因此，通过这种方法，如果结构约束为<**section**>，那么所有在字典中定义为与其同义标签的元素，就与<**section**>类元素没有区别。如果这些元素的内容符合要求，那么它们也可以作为答案返回。

第二种方法是结构增强（structure boosting），这种方法中首先忽略查询中的结构约束而对元素进行评分，接下来通过元素在多大程度上符合结构约束来提高元素的得分。元素的结构与查询结构进行对比，根据符合的程度来产生结构得分。例如，可以比较路径 [333，1579]，或者比较路径上的标签 [1626]。下面我们给出一个例子。

考虑使用 NEXI 查询语言（参见 13.6.3 节）的内容-结构查询 q，用来查找包含关于 wine 的文章中的关于 Patagonia 的节：

```
//article[about(., wine)]//sec[about(., patagonia)]
```

这个查询可以切分为两个独立的子查询 q1= //article[about(.,wine)]以及 q2= //sec[about(.,patagonia)]。

两个子查询分别进行处理。我们先来处理 **q2**。根据表 13-1 中给出的项统计信息，将元素评分策略用于子查询 **q2** 中，我们可以到如表 13-4 所示的内容检索得分。需要注意的是，

它们与在表 13-1 中给出的 $P(\text{patagonia} \mid M_e)$ 值相对应。我们将这些基于内容的得分记做 *c_score*。表 13-4 中也给出了表示与 section 元素符合程度的每种元素类型的结构得分，记做 *s_score*。在我们的例子中，section 元素记为<**sec**>，而其他元素类型包含<**para**>、<**sub-**
563 **sec**>、<**sec**>和<**article**>。通过这些得分可以反映出，<**subsec**>与<**sec**>的符合程度比<**para**>与<**sec**>的符合程度要好。我们假设基于内容的得分可用通过如下的方法对 **q2** 进行增强：

$$b_score(e,q2) = 0.8 \times c_score(e,q2) + 0.2 \times s_score(e,sec)$$

这意味着我们认为内容得分的重要程度要大于结构得分。因为 **q2** 是要查找 section 元素，那么元素 *e* 通过 *s_score*(*e*, *sec*) 来增强。表 13-4 给出了增强后的结果 *b_score*。例如，对于 para111，*c_score*(*para*111, *q*2)=0.327，*s_score*(*para*, *sec*)=0.4，那么 *b_score*(*para*111, *q*2)=0.8×0.327+0.2×0.4=0.341。

表 13-4 对于子查询 *q*2 的内容检索得分 *c_score*，内容-结构得分 *b_score*，以及结构得分 *s_score*

元素	*c_score*	*b_score*	*s_score*	元素	*c_score*	*b_score*	*s_score*
para111	0.327	0.4	0.341	sec1	0.170	1	0.336
para113	0.261	0.4	0.288	article	0.111	0.7	0.378
subsec11	0.233	0.6	0.293				

对于查询 **q1**，为了简化起见，我们只考虑将 article 元素作为结果，即我们假设对结构约束进行严格的解释。这就相当于对于所有非 article 元素 *e*，*s_score*(*e*, *article*)=0。

现在对于完整的检索 **q**，我们需要将通过检索 **q2** 得到每个元素的增强得分，与通过检索 **q1** 得到的篇章增强得分进行合并。可以按照如下方式进行定义：

$$s_score(e,q) = b_score(e,q2) \times b_score(article,q1)$$

根据表 13-1，我们可以得到 *c_score*(*article*, *q*1)=0.122（它与 $P(\text{wine} \mid M_{article})$ 相符合)，它与 *b_score*（*article*, *q*1）相同。图 13-7a 给出了排序后的元素列表。

a)	b)
0.0459 article	0.0588 para111
0.0415 para111	0.0300 para112
0.0409 sec1	0.0252 para22
0.0373 subsec11	0.0052 para113
0.0350 para113	

图 13-7 a）内容-结构查询排序结果；b）去掉重复后的结果

这里，一个重要的问题就是确定不精确匹配的实际程度（根据结构约束可以提供多少提示）。在我们的例子中，
564 这个可以变换为我们如何设置结构得分*s_score*。

这里介绍的技术都在 INEX 环境下进行了评价，其中元素是否相关都是仅基于内容进行评价的。此外，通过仔细研究这些实际的结构约束，可以发现它们并不是真正的提示 [1601]。虽然一个查询要求查找一节，但是这并不能说明，在相关数据集中，节元素就一定比符合查询内容要求的其他类型元素好。另外，结构信息并不是总能提高检索性能，除非在很靠前的排序中。这个多少令人有些失望的结果，也可能是由于评价方法的原因造成的。

13.4.4 去除重叠

XML 检索系统的目的是对于用户给定的查询，返回最相关的元素。当一个元素对于给定的查询被估计为相关的时候（不论通过本章介绍的什么 XML 排序策略)，它的祖先元素也可能会被估计为与查询相关（虽然程度不同)。此外，这个元素很可能包含一些也会被估计为相关的后代元素（虽然是在不同的程度上)。这是由于 XML 文档的嵌套结构所导致的，

一段相同的文本会出现在一个路径上的不同元素中。因此，元素自身、它的祖先以及一部分后代元素都会包含在结果列表中，最终会导致大量的重复信息返回给用户。在 13.4.3 节的很多排序结果中可以清楚地看到这一点。

返回重复的信息（即重叠的元素）已经证明会使用户感觉混乱 [1592]。在用户不喜欢或者不希望看到同样的信息多次出现的检索场景中，需要确定哪些相关但是重叠的元素是要返回的，这就是我们要讨论的问题。

第一种方法是直接从 XML 检索系统返回的初始排序列表中将重叠元素去除。最常见的去除方法（称为蛮力过滤，brute-force filtering）是在排序列表中选择得分最高的元素，然后删除它的祖先和后代中得分较低的元素。这个过程迭代进行，它依赖于检索的排序策略，即在重叠的元素中，哪些元素应该被赋予比较高的排序。

我们将这种方法用于对图 13-5a 进行排序，以生成如图 13-7b 所示的不包含重叠的结果。如果我们从如图 13-6a 所示的排序列表开始，那么会得到不同的结果。但是请注意，所有这些最初的排序都不适合产生最相关但非重叠结果这一目的。因此，很多方法在产生非重叠结果时考虑了文档的实际结构，如 [389，1099，1127，1291]。

例如，在 [1099] 中介绍的方法，在对元素评分时考虑到它们在 XML 文档树结构上的分布。例如，如果一个元素的很多后代元素都被检索出来，但是它们均匀分布在相应树结构中，并且与其父结点得分相近，那么会选择父元素——因为，可以从选择出来的元素访问它的相关后代结点；其他情况下会选择它的后代元素本身来进行处理。 565

对于我们给出的样例，从图 13-5a 所示的排序出发，这种方法会选择由 subsec11 和 para22 构成非重叠的结果。因为从 subsec11 可以访问到所有的段落，而 para22 是 sec2 中唯一包含相关信息的元素，只返回这一个段落更加合理。

总之，在去除重叠的技术中，直接考虑文档树结构的方法会比不考虑树结构的方法要好。然而，因为重叠去除过程是在查询阶段进行的，所以处理速度是一个问题，这就要求方法不仅有效，而且还要高效。原始结果列表对重叠去除策略影响的研究是一个吸引人的问题。很多迹象表明，好的初始结果列表会带来更好的非重叠结果列表。

13.5 XML 检索评价

随着 XML 成为结构化文档的标记语言，以及与 TREC（参见 4.4.1 节）等价的、用于 XML 检索性能评价的 INEX 评测的创建，结构化文本检索的研究显著增多。

在传统的信息检索评价中，文档被看做是独立和分离的单元。然而，XML 检索准许检索文档的各个部分，而同一篇文档中的各个部分不能看做是独立的单元。此外，当准许检索任意元素时，我们必须还要进一步考虑元素间的重叠。原因是，如果我们检索了一个完整的、包含很多段的节作为一个元素，之后我们又检索了包含在这一节中的某一段作为第二个元素，那么用户就会得到重复的信息。这就意味着检索得到的元素不能当做分离的单元。

根据这些结构化文本所特有的性质，对 XML 检索系统的评价在构建测试文档集时，其评测范式所遵循的标准，需要考虑固有的结构约束，即元素间的结构关系。INEX[⊖]（The Initiative for the Evaluationof XML retrieval）建立了包含大规模测试集和相应测度的基础

⊖ INEX 是由 DELOS 举办，它是一个专注于数字图书馆领域的欧盟网络。关于 INEX 在 2007 年以前的信息可以访问 http://inex.is.informatik.uni-duisburg.de/，2008 年之后的信息可以访问 http://www.inex.otago.ac.nz/。

设施，用于评测 XML 检索性能，这就是我们这里要讨论的。

13.5.1 文档集

在 2004 年之前，INEX 使用的文档集包含 12 107 篇使用了 XML 标记的文章，它们是从 12 本杂志和 IEEE 计算机学会的 6 本期刊上收集来的，时间跨度是 1995—2002 年，文档集大小是 494MB，包含 800 万个元素。平均下来，每篇文章包含 1532 个 XML 结点，每个结点的平均深度是 6.9。2005 年，文档集又从 IEEE 计算机学会的出版物中进一步扩充了一些。总共 4712 篇自 2002—2004 年的新文章加入了进来，扩展后总共包含 16 819 篇文章，文
566 档集大小也扩展到 764MB，包含 1100 万个元素。

从 2006 年起，INEX 使用了新的文档集，它是从 Wikipedia[⊖] 上抽取的英文文档 [493]。这个文档集包含使用 XML 标记的 659 388 篇从 Wikipedia 中选出来的文章，文档集的大小超过 60GB（不包含图片的大小是 4.6GB），其中包含 5200 万个元素。文档集的结构与 IEEE 文档集的结构基本相同。每篇文章平均包含 161.35 个 XML 结点，每个元素的平均深度是 6.72。这个文档集包含更丰富的标签类型（1241 个不同的标签，而 IEEE 文档集中只有 176 个），并且包含大量的交叉引用（cross-reference）（标记为 XLink）。

13.5.2 主题

在 13.6 节中我们提到，一个 XML 查询既可以包含内容也可以包含结构。将这个因素考虑进去，INEX 定义了两种类型的主题：

- **内容（Content-Only，CO）主题**，仅将信息需求进行描述，忽略文档的结构，在某种意义上，与应用于信息检索测试集的传统主题类似。在 INEX 中，这类主题的检索结果包含多种多样的元素，元素属于 XML 文档结构的不同层次。
- **内容-结构（Content-and-Structure，CAS）主题**，这类主题对所需元素的内容和结构信息进行描述。这些描述可能是指特定的元素内容（例如，要返回的元素必须包含有关特定主题的节），或者可能要指定返回元素的类型（例如，要返回的节）。

CO 和 CAS 主题反映了对文档集结构具有不同层次理解的用户搜索相关信息的过程。没有文档结构知识或者不使用结构知识的用户采用 CO 主题。搜索 XML 文档集的大多数用户属于这一类。CAS 主题适合于希望利用文档结构知识来提高检索质量的用户。CAS 主题更符合专业用户，例如图书馆员或专利检索人员等[⊜]。

与 TREC 类似，INEX 主题由标题、描述和叙述三个标准域组成。对于 CO 主题来说，标题包含一组查询项。对于 CAS 主题来说，标题使用 NEXI 来表示，它是基于一种基于路径的 XML 查询语言（参见 13.6.3 节）。

2005 年，CO 主题扩展为仅考虑内容＋结构（Content-Only＋Structure，CO＋S）的方式。CO＋S 主题包含一个 CAS 标题（**<castitle>**）域，它不仅表示 CO 主题中在**<title>**域中描述的相同信息需求，还利用结构约束加入了更多的知识。CAS 主题使用 NEXI 查询语
567 言表示。图 13-8 给出了一个 CO＋S 主题的例子。

⊖ http://en.wikipedia.org。

⊜ CAS 主题也可以用来作为相关反馈处理的结果，生成不仅包含查询项，而且还包含结构约束的新查询（例如 [1436]）。

```
<inex_topic topic_id="231" query_type="CO+S">
<title>Markov chains in graph related algorithms</title>
<castitle>
   //article//sec[about(., +"markov chains" +algorithm +graphs)]
</castitle>
<description>Retrieve information about the use of markov
chains in graph theory and in graphs-related algorithms.
</description>
<narrative>I have just finished my Msc. in mathematics, in
   the field of stochastic processes. My research was in a subject
   related to Markov chains. My aim is to find possible
   implementations of my knowledge in current research. I'm mainly
   interested in applications in graph theory, that is, algorithms
   related to graphs that use the theory of markov chains. I'm
   interested in at least a short specification of the nature of
   implementation ({\em e.g.} what is the exact theory used, and to
   which purpose), hence the relevant elements should be sections,
   paragraphs or even abstracts of documents, but in any case,
   should be part of the content of the document (as opposed to,
   say, vt, or bib).
</narrative>
</inex_topic>
```

图 13-8 INEX 2005 测试集中的一个 CO+S 主题

13.5.3 检索任务

XML 检索与传统的扁平文档检索最大的不同就是，XML 检索系统不仅要对每个元素与查询的相关程度进行评分，而且还要确定返回给用户的、合适的元素颗粒度水平。在 INEX中，相关元素的定义是在合适的可粒度下，如果它论述了用户查询所需要的所有主题（对查询来说是穷尽的），并且它没有论述其他主题（对于查询来说它是专一的）（我们在 13.5.4 节中介绍穷尽性和专一性）。

直到 2004 年，在 INEX 中，XML 检索系统的任务都是对于用户查询返回最相关的 XML 元素而不是整个文档，即最专一并穷尽的。换句话说，XML 系统需要返回包含尽可能多的相关信息和尽可能少的不相关信息的部分。在这样的一般性任务下，可以定义如下两个子任务：

- **CO 子任务**，使用 CO 主题，有效的 XML 检索系统能够得到最专一的元素，并且只包含与所需主题相关的内容。
- **CAS 子任务**，使用 CAS 主题，有效的检索系统能够检索出与主题需求相关的、最专一的文档部分，并且以严格或者近似（模糊）的程度匹配查询中的结构约束。它包含两个子 CAS 任务： 568
  - □ **SCAS 子任务**（严格的内容-结构），对于结构约束进行严格的解释。2002 年和 2003 年，对这个子任务进行了研究。
  - □ **VCAS 子任务**（模糊的内容-结构），对于这个任务，XML 检索系统的目标并不是返回严格符合结构约束的元素，但是路径约束在查找相关内容时会作为提示。

    这个子任务是在 2003 年提出来的，接下来从 2004 年开始，CAS 任务只包含这个子任务。指定信息需求不是件容易的事情，特别是在 XML 文档包含大量标签名（元素类型）的情况下，因此 SCAS 被认为是不太切合实际的任务。

在 CO 或 CAS 等一般性任务中，并没有考虑检出元素间的实际关系，许多系统返回了相互覆盖的元素（即嵌套元素）。例如，在 INEX 2004 CO 子任务中，前 10 名的系统包含 70%～80%的重叠元素。这与有效性的评价方法（参见 13.5.5 节）是十分相关的，如果一个方法试图更聚焦一些（例如两个嵌套的元素，只返回最专一的那个），那么它的得分就会变差。因此，2005 年定义了如下两个子任务：

- **“聚焦”子任务**，它的目标是使得系统在给定文档的一条路径中发现最穷尽和最专一的、包含相关内容的元素，作为最合适的检索单元返回给用户。不允许有重叠元素返回。
- **“全面”子任务**，它用来作为 INEX 的初始任务，即到 2004 年大多数系统都完成的部分[⊖]。

有了这两个子任务，就有可能更好地对 XML 检索方法针对给定查询所估计的元素相关性进行评价，而 XML 检索方法的目标就是对于给定的主题，找到合适颗粒度的相关内容返回给用户。这两个子任务都可以使用 CO 和 CAS 主题。对于后者，结构约束中还引入了模糊解释，即 VCAS 子任务。

13.5.4 相关性

在主要词典中，相关性的含义定义为“与手头的事情有关”。在传统的文档检索中，相关性通常理解为检出的条目与用户查询间的联系。在 XML 检索中，检出的条目与用户查询间的关系更复杂，还需要考虑文档的结构。因为检出的元素可以在所有颗粒度等级上，所以一个元素和它的子元素都可以与给定的查询相关，但是子元素比它的父元素可能更聚焦在查询的主题上，父元素可能包含其他不相关的内容。在这种情况下，子元素比它的父元素更适
569 合作为检出的元素，因为它不仅与查询相关，而且是特别针对查询的。

为了适应专一性，INEX 将相关性定义为如下两个方面：

- **穷尽性**，用来衡量一个元素反映用户需求主题的完善程度。
- **专一性**，用来衡量一个元素是否只聚焦于所需主题的程度（而不是其他无关主题）。

此外，多级尺度也是需要的，以便明确表示一个元素与其子元素相比，讨论某个主题的穷尽程度。例如，一节中包含两个相关段落，则可能认为比其中任何一个单独段落都更相关。用二值表示的相关度不能反映这种不同。和穷尽性一样，专一性也需要用多级尺度进行描述，这样可以奖励有能力选择最合适（精确）大小元素的检索系统。例如，能够确定书中哪一节相关的检索系统，比只能确定哪一章相关的系统要更有效些。因此 INEX 对于穷尽性和相关性使用了四级相关性尺度：“不相关”、“边缘”、“相关”和“非常相关”。穷尽性和专一性的结合，以及它们的分级尺度，使得有可能发现那些既穷尽又专一的相关元素，因此可以将最合适的单元返回给用户。

评审人员作为 INEX 的参与者，每年都通过在线相关性评估工具来提供相关性评估 [1278]。与 TREC 一样，采用聚合方法（pooling method）选择需要被评估的元素。在 2003 年和 2004 年，在线评估工具会显示文档及其要评估的元素。评审人员对每个元素确定一个相关度值。有些实施规则确保相关元素会被评估（例如，如果一个在池中的元素被确定为相关元素，那么它的父元素和后代元素都会被加入到池中，因为它们很可能是相关的，不过是在不同的程度上），也确保相关性元素的赋值在合理的范围内（例如，一个元素被确定为相

⊖ INEX 中还定义了一些其他的任务，感兴趣的读者可以参考 [968]。

关，那么它的穷尽性程度一定不会大于它的父元素）。但是到了 INEX 2004，实施规则已变得过于苛刻，成为评估的阻碍。

由于引入了简化的评估程序，这种情况在 2005 年得以改变。需要评估的元素依然会显示给评估人员，但是评估人员只需要找出相关的段落，而不是对每个元素的相关性进行判断。同时，也引入了加亮方法（见图 13-9），使得相关性评估更加自然和非侵入式，从而得到了更高质量的评估结果 [1279]。

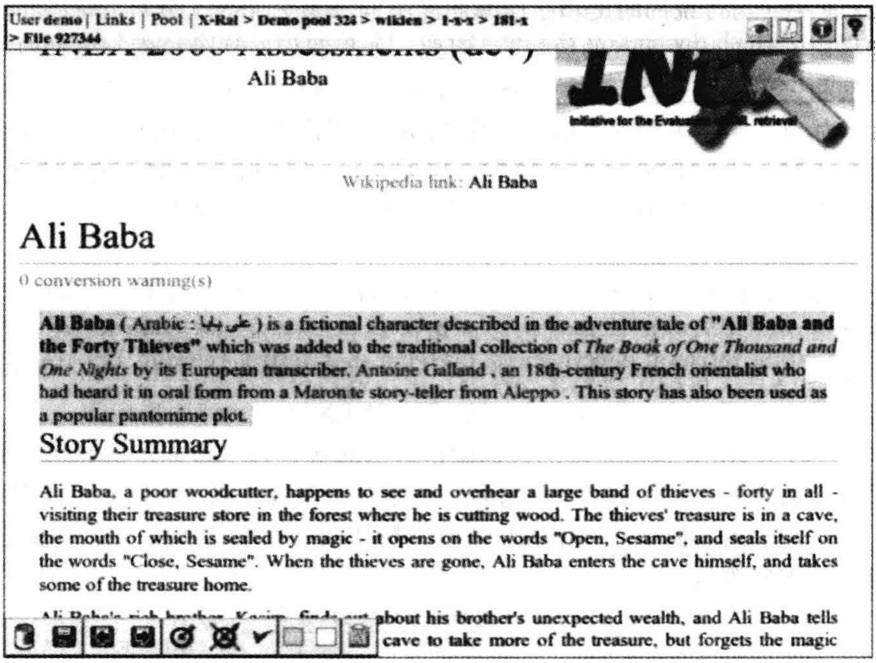


图 13-9 INEX 2006 评估界面

评估过程包含两个阶段。在第一个阶段中，评估人员对仅包含相关部分的文本块进行加亮。接下来，根据 XML 元素中与标注部分的相关部分的多少，对于专一性维度自动进行 0～1的连续评分；完全加亮的元素赋值为 1，没有加亮的元素赋值为 0；其他元素的专一性得分定义为加亮文本（即相关信息）和元素大小的比例（以字符为单位）。在第二个阶段中，针对所有的属于加亮段落的元素（以及它们的父元素），评估人员都会对它们的穷尽性进行评估。穷尽性属性则使用四等级尺度。 570

通过对 INEX 2005 的结果进行广泛的统计分析 [1226]，我们看到如果忽略掉穷尽性维度，对检索结果的性能比较没有太大的变化。因此，在 INEX 2006 中放弃了穷尽性，从此开始对相关性的评测仅考察专一性维度（与 2005 年采用同样的方法）。因此，评估人员在评估过程仅需完成第一阶段，这进一步提高了评估结果的质量。

经过多年的实验，对 XML 检索来说，用加亮过程来收集评估信息，足够可靠和完备。在未来的 INEX 评测中，研究人员会针对更大范围的聚焦检索任务（包括段落检索、问题回答、元素检索等）进行评测。观察这一评估机制的稳定性将会是一个有趣的课题，因为对不同的聚焦检索任务采用统一的评估机制会带来很大的好处 [1599]。

13.5.5 测度

对 XML 检索进行有效性评价需要考虑元素间的依赖关系。与传统的文档检索不同，用

户在 XML 检索中可以访问结果集中的其他结构相关的元素。因此他们可能需要通过浏览或者滚动（如何操作及范围取决于界面）找到其他的相关信息。这启发了在评价中对所谓的微小失误（near-misses）问题的考虑，它是在评价中用户可以通过它访问到相关内容的元素。另外一种方法就是忽略微小失误，这样会造成评价标准过于严格。TREC-8 中的 Web 任务，
571 通过在评价检索性能时，考虑链接到相关内容的网页，也得到了类似的调查结论 [723]。

从 2002—2004 年，INEX 使用 *inex_eval* 评测程序 [660]，它将 precall[1328] 测度引入到 XML 元素中。与传统的精度和召回率测度不同（参见 4.3.1 节），*inex_eval* 是基于计数机制，即检出元素与相关元素的数量⊖。因此，如果我们在评价检索有效性时考虑微小失误，那么返回相关但是有重叠元素的系统，比返回相关但是没有重叠元素的系统有更高的有效性。例如，如果一个相关段落和包含它的节都被检索出来，那么这两个元素都会被认为相关元素（实际上是因为段落相关才使得节也相关的），这就增加了检出的相关元素的数量。因此，尽管并未检出新的相关信息，但返回重叠相关元素的系统可以得到更高的有效性得分 [886]。

目前已经提出了太多对于 XML 检索的评价方法（例如 [483, 631, 885, 1253, 1276]）。在本节中，我们介绍一种从传统的精度和召回率延伸而来的方法，它将在 INEX 2005 的加亮评估过程中得到的知识考虑进来。这些测度最初是由 Pehcevski 和 Thom[1253] 提出的，最后发表于 [867]，从 2007 年开始成为 INEX 的正式评测指标。

我们回忆一下，XML 检索系统的目标就是返回那些包含尽可能多的相关内容和尽可能少的非相关内容的元素。利用加亮评估过程，这个过程就转变成了返回尽可能多的加亮（相关）内容，尽可能少的非加亮（不相关）内容。精度和召回率的经典定义（参见 4.3.1 节）可以按照这个要求修改为：

$$\text{精度} = \frac{\text{检出相关信息的数量}}{\text{检出信息的总数量}}$$

$$\text{召回率} = \frac{\text{检出相关信息的数量}}{\text{相关信息的总数量}}$$

这里我们不统计相关项（元素）数目，而是统计检出的相关（加亮）的文本。

更正式地，用 $hlength(e)$ 表示对于已给定的主题，在元素 e 中包含的加亮文本的字符数（如果元素中没有加亮的文本，那么 $hlength(e)=0$）。$length(e)$ 表示元素 e 中包含的全部字符数，$Trel$ 表示对于给定的主题，在文档中所有相关（加亮）信息的总字符数。$erank(i)$ 函数返回排序为 i 的元素。前 r 位精度，记做 $P@r$，表示排名前 r 位的元素中检出
572 相关信息的比例：

$$P@r = \frac{\sum_{i=1}^{r} hlength(erank(i))}{\sum_{i=1}^{r} length(erank(i))} \tag{13-5}$$

这个定义确保为了在排位 r 上得到高的精度，排在其之前（包括第 r 位）的元素必须包含尽可能少的不相关内容（即对每个 $erank(i)$ 最大化 $hlength(erank(i))$）。

前 r 位召回率，记做 $R@r$，表示到第 r 位为止，相关信息被检出的比例：

$$P@r = \frac{1}{Trel} \times \sum_{i=1}^{r} hlength(erank(i)) \tag{13-6}$$

⊖ 对非二元相关性，将每个检索的元素的相关值在计数中增加。

这个定义确保为了在排位 r 上得到高的召回率，排在其之前（包括第 r 位）的元素必须包含尽可能多的相关内容。

Trel 的定义取决于是否准许返回重叠元素，即全面的还是聚焦的检索子任务（参见 13.5.3 节）。对于全面子任务，*Trel* 表示所有元素中加亮字符的总数；对于聚焦子任务，*Trel* 表示所有文档中加亮字符的总数。两者之间的区别是，对于前者，相互重叠的文本（重叠的相关元素中包含的）都用来计入加亮相关部分的总数；对于后者，使用的是非重叠文本。

这种测度的一个最大的好处是，对每一个查询，它们不需要从全部相关文档中构造理想的（非重叠的）相关元素集。其他一些为了克服 *inex_eval* 缺陷而特别制定的 XML 评价测度则需要这样的信息。从文章 [882] 报告的实验结果可以看出，理想集的选取方法直接影响检索性能的评价。

对于信息检索的精度和召回率等传统测度，固定召回率水平的精度和平均精度等其他测度也进行了定义（参见 4.3.2 节）。这些测度的可解释性与从标准精度和召回率派生的测度类似（例如，P@10%，P@20%，…，P@100%，MAP)，这些测度在信息检索研究领域已经被很好地确定和理解。

最后，因为 INEX 合并了其他种类的聚焦检索任务，所以这些定义在加亮文本上的测度，也可以用来评价其他任务，比如用于 INEX 2007 新设置的段落检索任务 [606]。 573

13.6 查询语言

在检索非结构化文本时，用户查询的表达能力自然就受到了一定的限制，因为用户只能询问文档关于什么或是包含什么单词。在结构化文本中，有了查询语言的帮助，用户有了表达更精确查询的能力，例如，“我想找一个讨论 penguin 的段落，它在一个标记了 South Pole 的图片旁边”。这里，“penguin” 和 “South Pole” 是对要检索的内容给定的约束条件，而“段落”和“图片”是对要检索的结构给定的约束。我们现在要讨论的查询语言就是 XML 和结构化文本检索必不可少的组成部分。

13.6.1 特性

文本检索查询语言的特性可以根据要表达的约束的种类，归纳为三个主要的组：内容约束、模板匹配约束和结构约束。

1. 内容约束

这些约束与内容方面的信息需求相关，有多种类型的内容约束存在：

- **单词**：一个或者多个用来指定返回的文档片段应该包含或者接近的查询单词。对于大多数信息检索系统而言，这是经典的输入。
- **上下文**：单词在文本中的位置，例如构成一个短语，或者在一个给定的距离内（例如，“information and retrieval with distance 4”）。
- **权重**：单词或者上下文约束在文档片段中的重要性。例如，“0.6 penguin 0.2 swim” 的含义是当判断文档片段是否作为结果返回时，“penguin” 比 “swim” 要重要，“+penguin swim” 意思是返回的文档片段必须包含或者关于 “penguin”。
- **布尔**：以上所有的约束都可以使用布尔运算合并。例如，“0.6 penguin or (south pole with distance 4)”。满足布尔表达式的文档片段可以作为结果返回。

在（传统）数据库中，处理查询的内容约束（大多数情况下）会得到一个未排序的文档

片段列表，然而在信息检索里，这个列表要经过排序。此外，在数据库中，返回的文档片段中通常要求单词必须包含在其中，但是在信息检索中，包含被关涉性（aboutness）替换了，即返回的文档必须与（单词所描述的主题）单词相关。

2. 模板匹配约束

这些约束准许检索的文档片段符合特定的模板，例如字符串、前缀、后缀、子串或者正
574 则表达式等。

3. 结构约束

结构约束准许指定结构方面的信息需求。它主要包含三种类型［47］：

- **目标结果**：如果已经知道所需结果的结构，那么用户可以通过指定具体的结果结构来满足他们的要求（例如，用户从科研论文集中查找“摘要”部分）。
- **支持条件**：结构可以用来指定非最终结果的结构约束。例如，一个用户查询“从摘要和 red wine 有关的文章中找出有关 Chile 的章节”，这里包含两个结构约束：一个对目标结果节（关于 Chile）的要求，另外一个是摘要（关于 red wine），提出了另外的结构约束。后者就是支持条件。
- **结果构建**：用户想得到的查询结果，不仅仅是已经存在的片段（出现在文档结构中），而且还希望结果可以由一个或者多个文档的多个片段构造。例如，用户可能希望得到“一节的题目，以及这一节的第一段和最后一段组成的片段”。

用来访问 XML 文档的查询语言，就是为了全部或者部分满足上面列出的这些内容和结构约束。需要注意的是，表达能力的增强就意味着，将用户信息需求转变为正确的查询和处理这些查询时的复杂性的增加。

13.6.2 XML 查询语言分类

XML 查询语言可以分为内容（content-only）和内容-结构（content-and-structure）查询语言两种。

1. 内容查询语言

内容查询语言使用内容约束来表达用户的信息需求。最简单的形式是由一些单词组成，这长期以来一直是传统信息检索的标准输入形式。这种查询语言适用于用户在表达信息需求时不知道或者不关心文档结构的 XML 检索领域。虽然仅指定了内容方面的信息需求，但 XML 检索系统依然需要决定哪些片段，即哪一个颗粒度等级上的 XML 元素最适合作为要提供的信息。

2. 内容-结构查询语言

这种类型的查询语言可以方便用户指定在内容和结构方面的信息需求。大多数 XML 查询语言的研究也都是朝这个方向发展的。13.6.1 节列出了这些查询语言所具有的内容和结构特性。这里我们主要讨论这些结构特性是如何用查询语言描述的，但是具体的句法不是我
575 们这里的重点。

内容-结构查询语言包含三种主要的类型，标签语言、路径语言和子句语言，其复杂性和表达能力从标签语言到子句语言依次增加。从用户角度来看，这种表达能力和复杂性的增加，通常意味着内容-结构查询语言难于书写。但是它们对于高级用户在特定的领域十分有用，例如专利检索和基因检索。

（1）标签查询

这类查询准许用户对查询中的单词标注一个标签名，用于指定目标结果的结构约束。例

如，对于信息需求“检索关于 red wine 的节”可以表示为如下方式：

```
section: red wine
```

标签查询语言可以将基本的结构约束引入进来，但是与路径查询语言和子句查询语言相比缺乏表达能力，这是因为标签查询语言不包含支持条件和结果构建。XSEarch[405] 是标签查询语言的一个例子。

(2) 路径查询

这类查询基于 XPath 语法，在查询中封装文档的结构。路径查询语言的例子包括 XPath 1.0[⊖]、XIRQL[603] 以及 NEXI[1602]。例如，对于信息需求“在关于 Chile 的文档中检索有关 red wine 的节”用 NEXI 表示为：

```
//document[about(., Chile)]//section[about(., red wine)]
```

路径查询可以表示目标结果（上例中的“section”元素）和支持条件（“document”关于“Chile”）。

XIRQL 准许对结构约束赋以权重，例如 XIRQL 查询：

```
//section[0.6 .//* $cw$ "Chile" + 0.4 .//section $cw$ "wine"]
```

要求检索节，该节所包含的元素（任意类型、任意级别）包含单词“Chile”，还要求节元素包含单词“wine”，其中第一个结构约束比第二个约束在估计哪些节可以作为结果时要重要。

需要注意的是，任何标签查询，例如，`section:red wine`，可以用路径查询进行重写，例如，用 NEXI 可以重写为：

```
//section[., about(red wine)]
```
576

此外，所有内容查询也可以用路径查询表示，例如下例中的 NEXI 查询语言：

```
//*[., about(red wine)]
```

它要检索关于“red wine”在任意颗粒度水平下的元素。

(3) 子句查询

这类查询使用嵌套子句来表达信息需求，与数据库中的 SQL 语言（结构化查询语言）很类似。在 XML 检索中，最突出的子句查询语言是 XQuery[⊖]。典型的子句查询由三个子句组成：“for”子句指定支持条件，“where”子句指定内容约束，“return”子句指定目标片段或者重组新的片段作为结果。在“for”和“return”子句中，使用 XPath 表达式来指定文档的结构。对于信息需求“检索标题为 penguins 的文档的节”，用 XQuery 可以如下表示：

```
for $x in /document/section
    where $x/title=penguins
    return $x/section
```

全文 XQuery[42] 扩展了 XQuery，具有强有力的全文检索操作，包括 13.6.1 节中介绍的上下文约束（例如邻近距离）和排序函数等。

13.6.3 XML 查询语言样例

在本节中，我们将介绍四种内容-结构查询语言。在 13.6.2 节中我们提到，XML 内容查询语言可以使用与扁平（非结构化）文本检索一样的方法指定，因此，本节中我们就不再详细讨论。我们这里介绍两种路径查询语言，XPath 和 NEXI，以及两种子句查询

⊖ http://www.w3.org/TR/xpath。

⊜ http://www.w3.org/TR/xquery/。

语言，XQuery 和全文 XQuery。这些查询语言对目前 XML 查询语言的发展提供了很好的概述。

1. XPath

XPath（XML 路径语言）是 W3C 定义的查询语言。它的主要目的是用来访问或者浏览 XML 文档。此外，XPath 提供了基本的字符串、数字和布尔操作功能。XPath 的第一个工作草案（XPath 1.0）在 1999 年 7 月发表，1999 年 11 月获得了推荐[⊖]。

XPath 中最重要的表达式类型是定位路径（location path），它包含了在 XML 文档中的
577 一系列导航步骤。book/publisher/@isbn 是一个定位路径，其中 book 和 publisher 分别是导航到（或者选择）"book"和"publisher"子元素的步骤，@isbn 是导航到"isbn"属性的步骤。所有这些步骤间用"/"分割，表示定位路径选择直接在 publisher 元素下的 isbn 属性，而它直接在 book 元素下。publisher 元素作为 book 元素的子元素。"/"对应于所谓的子结点和双亲在 XPath 轴中的一步。

导航步骤也可以由"//"分割，它表示定位路径在进行下一步前，导航到当前元素和它的所有后代元素。例如，book//title 导航到所有直接或者间接在 book 元素下的 title 元素，其中 //title 会选择文档中的所有 title 元素。

特殊的步骤包括自身步（self step）记做"."以及双亲步（parent step）".."。例如，.//book 返回当前结点中包含的任何 book 元素，而 ../publisher 返回当前结点的父结点所包含的 publisher 元素。另外，XPath 使用通配符"*"和"@*"来导航到任意名字的元素和属性，例如 book/* 以及 book/publisher/@* 。

在每一步中，可以在"["和"]"之间指定断言，选择（导航）的结点必须满足这些断言。例如，XPath 表达式 //book[@year= 2002]/title 选择的是包含且仅包含发表于 2002 年的书的题目。标准比较操作符=、!=、<和<=都可用于断言。存在断言用于判断一个路径表达式是否返回非空结果。例如，XPath 表达式 //publisher[city] 选择给定城市的出版社。最后，位置断言用来根据一个元素在文档树中的位置进行导航。例如 //publisher/country[1]/city 选择每个出版社所在的第一个国家的城市（我们假设一个出版社设在多个国家）。比较和存在断言可以用"and"和"or"操作以及"not()"函数进行合并。例如，not(@year= 2002)。

布尔函数 contains() 是 XPath 中进行面向内容的 XML 检索的一个重要函数，它的输入是两个字符串变量，输出是 true，表示第一个字符串包含第二个字符串；否则，输出 false。这个函数可以用来检查元素的文本（作为第一个变量）中是否包含了指定的字符串（作为第二个变量）。它的输出不是一个排序的元素列表，因此 XPath 不是一个可以直接应用于面向内容的 XML 检索的查询语言。然而，XPath 可以应用于其他 XML 查询语言或者已经启发了其他 XML 检索语言，其中一些准许对结果排序。

2. NEXI

Narrowed Extended XPath I（NEXI）查询语言 [1602] 是由 INEX 开发的、用于面向内容的 XML 检索评价的简单查询语言。NEXI 包含一个小的但是扩展的 XPath 子集，INEX参与者使用它来构造真实的内容-结构查询，用以构造测试集。NEXI 是基于 XPath 的，因为后者是 XML 界已经熟知的一种语言，所以如果 INEX 使用另外一种 XML 语言会
578 带来一些不利之处。

⊖ http://www.w3.org/TR/xpath。

NEXI 对于 XPath 的扩展是引入了一个新的函数，叫做 about()。XPath 中的 contains() 函数要求元素（其文本）包含给定的字符串内容，替换为 about() 函数，它要求元素与这个内容有关。about() 函数反映的是元素有可能与给定的查询（内容部分）相关，但是并不真正包含查询中使用的任何单词。

从 XPath 中选择一个小的子集的原因有两个方面。第一，从提交给 INEX 的查询分析可以看到，使用 XPath 的定位路径来表达结构约束会带来较高的语法和语义的错误率 [1227]。因此，所有对评价面向内容的 XML 检索的有效性不需要的定位路径都可以省略掉。例如，parent/child 导航步中的“/”是最有问题的，因为它会被错误地解释，所以被删除了。

第二，NEXI 是为了做检索评价而定义的。为了这个原因，不能使用位置断言，因为它们对有效性评测没有任何帮助。此外，所有目标元素都必须包含至少一个内容条件，即一个 about() 函数。这实际上是一个机械的过程，比如可以返回有关给定主题的节的标题。出于评价检索效果的目的，重要的是相关的节是否确实返回了。

例如一个 NEXI 检索项：

```
//article[about(.//body, "artificial intelligence")]//
          body[about(., chess) and about(., algorithm)]
```

目标结果是 //article//body。使用了几个内容约束，例如，about(.//body,"artificial intelligence")，有一个约束是目标结果。一个布尔操作用在 about(.,chess) 和 about(.,algorithm) 之间。

INEX 组织开发 NEXI 的目的，是为了 XML 检索性能评测而构建主题。因此 XML 检索任务中要对 NEXI 查询进行解释，其中解释是指 XML 检索系统通过检索模型实现对 about() 条件的解释，以及查询处理引擎对于结构约束的处理。

3. XQuery

XQuery 是使用 XPath 作为子语言的一种 XML 查询语言，但是增加了查询多个文档和合并检索结果到一个新的 XML 片段（结果构建）的能力。XQuery 是 W3C 工作组的成果，该工作组是在一个 XML 查询语言专题讨论会之后作为结果成立的㊀。XQuery 1.0 的推荐文档㊁发表于 2007 年 1 月㊂。 579

XQuery 的绝大多数特征都可以追溯到它的直接前身 Quilt[354]，它是为了查询异质数据集而构建的，并且借鉴了多种语言的设计。这些措施包含在 XPath 1.0（如前所述）和 XQL（面向文档的语言）中，其中路径表达式的概念引入到多层次结构化文档导航中。从 XML-QL（面向半结构化信息的语言）[494] 得到的想法是使用变量绑定来构建新的返回结果片段。较早的影响包括 SQL 的“select-from-where”子句来重组数据，以及合并、分组等操作，这些启发了 FLWOR 表达式的产生。此外，需要提到的是面向对象的语言（Object-Oriented Language，OQL）[345] 的影响，它贡献了由不同类别的可嵌套表达式组成的功能性语言。

XQuery 的核心表达式是 FLWOR 表达式。举例说明，考虑如下的 XQuery 表达式，其功能是根据名字排序，列出那些平均图书价格在 50 欧元之下的出版商：

㊀ http://www.w3.org/TandS/QL/QL98。

㊁ http://www.w3.org/TR/xquery/。

㊂ 2007 年 1 月，XQuery 1.0 和 XPath 2.0 的推荐文档一起发布。XPath 2.0 与 XQuery 1.0 采用了同样的数据模型，并且它在语义和语法上都是 XQuery 1.0 的一个子集。

```
for $pub in distinct-values (doc("pub.xml")//publisher)
let $a := avg(doc("bib.xml")/book[publisher = $pub]/price)
    where $a < 50
    order by $pub/name
    return <publisher> { $pub/name , $a } </publisher>
```

FLWOR 表达式以一个或者多个 for 和 let 子句开始，每个子句与一些变量（以 $ 开头）绑定。与 for 子句绑定的变量用于遍历表达式结果序列中的元素，与 let 子句绑定的变量用于遍历整个序列。可选的 where 子句指定选择条件，另一个可选的 order by 子句指定了排序准则。最后，return 子句指定了哪些结果会被返回。

在上面这个例子中，for 子句与变量 $pub 绑定，它以出现的先后顺序遍历文档“pub. xml”中所有的出版商元素。distinct-values 函数用来消除“pub. xml”中的重复。对于每个绑定的变量 $pub，let 子句绑定 $a 来计算来自于出版社 $pub 的书籍的平均价格。接下来，那些 where 子句为真的、符合条件的出版社元素被选择出来，即平均价格 $a 小于 50。结果根据 order by 子句给出的 $pub（$pub/name）中的出版社名字排序。最后，对于根据前面子句中产生的每一个绑定 $pub 和 $a 的结果，return 子句生成一个新的包含出版商 $pub 名字的出版商元素，并且包含平均价格 $a。若没有 order by 子句，结果则会根据出版商元素出现在“pub. xml”中的顺序进行排序。结果是新的片段，因为它们并没有出现在原始的 XML 文档中。

XQuery 对于 XML 检索是一个功能强大的查询语言，它可以看做是 XML 的 SQL。它是一个最适合于以数据为中心的 XML 检索语言。这是因为，它的文本检索内容有限，并且
580 它并不提供结果排序，而后者对于面向内容的 XML 检索十分关键。

4. 全文 XQuery

全文 XQuery[42] 是从 XQuery 扩展而来的一种 XML 查询语言，具有很强的文本搜索能力。例如，利用全文 XQuery，用户可以查找符合“包含单词‘growing’和‘wine’，并且它们的距离在三个单词以内，同时忽略掉单词 growing 的词干变化”的目标元素，这种需求不能用 XQuery 表达。此外，全文 XQuery 准许指定结果按照它们的相关度进行排序。

全文 XQuery 受到早期结构化文本检索语言的启发，例如 ELIXIR[372]、JuruXML[333] 和 XIRQL[603]。本节所基于的文档发表于 2008 年 5 月⊖。

增加的文本检索能力是通过一种新的 XQuery 表达式 *FTContainsExpr* 实现的，它作为一个普通的 XQuery 表达式完全可以与其他 XQuery 和 XPath 表达式组合。例如，下例中的 *FTContainsExpr* 表达式：

```
//book[./title ftcontains {"red" "wine"} all]//author
```

返回题目中包含指定单词“red”和“wine”的书籍的作者。

全文 XQuery 定义了文本检索的原语（即上下文内容约束），例如短语、词序、邻近单词等。它也允许在单词匹配时指定字母的大小写，还可以使用词干提取、同义词典、禁用词、内容模板匹配（使用正则表达式通配符），以及其他功能。例如，下例中的 *FTContainsExpr* 表达式限制了匹配的单词要在 6 个单词的窗口内：

```
//book[./title ftcontains {"red"  "wine"}
        all window at least 6 words]//author
```

下例中的 *FTContainsExpr* 表达式查找与单词“growing”匹配的多种形式，例如

⊖ http://www.w3.org/TR/xquery-full-text/。需要注意的全文 XQuery 的语法是可能已经演变了。

“grow”、“grows” 等：

```
//book[./title ftcontains "growing" with stems]//author
```

结果排序通过引入 *FTScoreClause* 表达式，可以指定评分变量。这些变量提供了访问 *FTContainsExpr* 表达式的评价得分功能。一个与信息检索搜索类似的例子如下：

```
for $b score $s in //book[./title ftcontains {"red" "wine"} all]
    order by $s descending
    return <book isbn="{$b/@isbn}" score="{$s}"/>
```

上面这个查询遍历所有题目中包含 “red” 和 “wine” 的书，其中变量 $b 将每本书的得分与得分变量 $s 绑定。两个变量用于返回书的 “isbn” 号和它的得分，按照相关性倒序排列。 581

全文 XQuery 并不是为了实现某个特定的评分方法而设计的，而是准许实际使用时按照自己的想法来处理。换句话说，上例查询中没有具体指定 $s 的值是如何计算的。每一个全文 XQuery 的实现方案都可以自己选择评分函数，只要生成的得分在 [0，1] 之内，越高的得分意味着越高的相关性。

全文 XQuery 具有数据和文档为中心的 XML 检索应用所需的所有特性。它就是为了这个目的而开发的。它可以实现用于对结果排序的评分函数。从面向内容的 XML 检索角度，全文 XQuery 可能会被许多终端用户认为过于复杂、难于掌握。但是它适用于一些需要专业用户介入的应用，例如医学领域或专利产业。此外，能够产生新结果的能力（XQuery 也具有）也对所谓的聚合检索 [966] 有帮助，即将从同一篇文档或者多篇文档的多个片段合并起来组成新的结果。

13.7 趋势和研究问题

结构化文本检索对于给定的查询，利用文档的内容和结构来确定最合适的文档片段作为答案。对于包含长文档或者包含了很多主题的文档（例如书籍、使用说明书和法律文件等）的信息资源库，它被认为是特别有帮助的，可以帮助用户直接定位到最有用的部分，而减少用户在一篇文档中查找相关部分的工作。

结合文本内容信息和文档结构信息的检索模型叫做结构化检索模型。我们讨论了结构化文档检索模型的结构化能力，也讨论了两种早期的这类模型。初步的结论是，模型的表达能力越强，其查询处理的效率就越低。因此，在根据给定的应用选择结构化文本处理模型时一定要仔细考虑。一条好的方针就是选择能满足应用所需功能的最高效的模型。

由于 XML 已经被采纳为结构化文本的格式，因此结构化文本检索现在已经变成了 XML 检索的同义词，XML 检索研究的起步早于 2002 年推出的 INEX。然而，面向内容的 XML 检索是在其之后才有了大的进步。对于查询处理、索引和检索，现在都还不能非常一致地肯定哪种方法或者哪些方法的组合可以得到最好的效果，因为确定元素与给定的查询是否相关，受到很多因素的影响，例如元素的大小、元素的类型、元素在文档树结构上的位置、结构相关元素的相关度、结构约束的解释，以及文档集的性质等。然而，我们可以假设，考虑元素的上下文、元素的大小，以及元素自身的内容（直接或者使用传播或聚合策略）来估计元素与给定查询之间的相关性对于 XML 检索是有益处的。另一个剩下的问题是内容-结构查询迄今所带来的检索性能改善是有限的 [1601]。 582

在本章中，我们没有讨论效率问题。许多检索策略需要引入复杂处理。不只是查询需要更多的处理（例如，由于它们的结构约束），而且可能检出元素的数量也比仅处理整篇文档要大得多。很多在 INEX 竞赛期间评估的方法都关注性能而忽略了效率。一些重要的贡献包

括 [601, 1578]。INEX 2008 启动的关注效率的任务[⊖]很受欢迎。

另外一个本章中没有讨论的关于 XML 检索的重要问题是用户界面。合适的界面对于越来越复杂的用户与检索系统间的交互是十分需要的，例如，为了能够表达内容-结构查询（例如 [1627]）和显示 XML 检索结果（例如 [868]）。

本章介绍的检索策略的目标是为了估计元素与给定查询之间的相关度。这并不一定是 XML 检索的最终任务。去除结果中的重叠是第二个检索任务。INEX 中研究的第三个任务是上下文相关性任务（relevant in context），它关注对于给定查询返回最相关的文档，并且在每一个文档中，找出最相关的元素。将结果列表中的元素按文档分组，比属于同一文档的元素分布在整个列表中，更适合用户 [195, 1591]。在 INEX 中研究的第四个任务是上下文最优任务（best in context）。其目的是找到一个唯一的文档入口，即 XML 元素，从这里开始阅读相关内容。这样的检索场景符合中等大小的文档集。

从 2007 年开始，INEX 包含了段落检索任务，其目标是找到大小合适的结果，并返回它的位置信息。段落检索自 90 年代中期在信息检索领域开始研究 [318, 734, 878, 1415, 1696]。段落检索任务的构想来自于 2005 年开始的评估过程，即对文档中的相关文档加亮。因此段落检索无非是试图找出这些加亮的片段，并对它们进行适当的排序。一个值得关注的问题在于，是先找到这些段落，然后再将它们映射到元素中去，还是直接将最好的元素作为查询的答案 [785, 814]。

一个受到越来越多关注的新范式是聚合搜索 [966]。在上下文相关性任务中，来自同一个文档中的元素组合在一起。没有什么能够阻止创建虚拟文档，即将来自于不同文档的元素智能地聚合成新文档。2008 年夏发布了关于聚焦检索和结果聚合的专刊论文征集。按照本章的描述，聚集检索这一术语体现在 XML 检索中的段落检索和问题回答 [1599]。事实上，作为段落检索和问题回答系统返回的段落和答案，也可以分别用做结果聚合。甚至是全文也可以使用，并且已经使用在主流搜索引擎，例如 Alpha Yahoo! 和
583 Google 的统一检索。

到目前为止，在 INEX 竞赛中开发的大多数 XML 检索方法都一直关注扁平的文本单元检索，例如元素和段落（近期才开始）。一个重要的遗漏是从结构化（层次化）文本中检索树结构，这种方法可以准许文档结构体现在结果中，同时解释文档树上的用户导航。例如，全文 XQuery 等 XML 查询语言包含对基于树结构约束查询的支持，可以指定基于树的结果。一些方法可以输出 XML 文档子树的排序序列，例如 XRANK[686]、XXL[1577] 和 XIRQL[603]。随着对树检索的关注度增高，对树检索的有效性进行评定的评价方法需求也提了出来 [26]。在写作本书时，INEX 竞赛连续举办了 7 年 [602, 609, 611, 610, 612, 606, 621]，今年是第 8 年。对 XML 进行评价是一个挑战，因为增加结构信息会带来很多复杂的问题，其中有些并不总能在开始时很好地预见到（例如，重叠问题和基于计数的度量，在 2 个维度上使用四级评分对元素始终如一评估的难度等）。此外，经费很有限，这些经费的用途包括支付评估人员费用等。这导致了研究本身的问题，例如，如何提高评估质量使得测试集可以重复使用 [1279]。通过 6 年的时间和一系列的变化（例如，XML 检索中相关性的定义，很多评价方法的提出），已经达到了一定的稳定性，使得现在的研究人员和开发人员可以全心专注于 XML 检索方法，能够了解在 XML 检索中哪些部分有效果，因为现在我们知道怎样评价和解释 XML 检索结果的有效性。

⊖ http://www.inex.otago.ac.nz/tracks/efficiency/efficiency.asp。

从 2002 年完成的综述 [488] 中可以看到，早期的 XML 文档语言在面向内容的 XML 检索中受到了很多限制。例如，它们不提供相似度检索，因此不可能对结果按照相似度排序。从而，出现了面向 XML 内容检索的查询语言。例如，XXL[1577]、ELIXIR[372]、XIRQL[603]、ApproXQL[1438] 以及本章介绍的 NEXI 和全文 XQuery。全文 XQuery 是为了符合 "XQuery 1.0" 和 "Xpath 2.0 Full-Text 1.0" 需求[1]和用例[2]而设计的。目前的规范是 W3C 的候选推荐。一旦某些条件达到后（例如，一个可以测试每一个独立特征的测试组件，本规范的最小一致性被至少两个独立的实现方案证明），这个文档打算作为 W3C 提议推荐提交。这个文档发表于 2008 年 5 月 16 日，对于开发和实现已经是足够成熟和稳定[3]。

XML 检索对于信息检索的所有领域都变得越来越重要。虽然我们这里没有讨论，但类似于 TREC，INEX 独立运行多个额外的任务来研究 XML 检索的不同方面和任务，包括交互、相关反馈、异质文档集检索、实体排序、自然语言查询和问题回答等。有兴趣的读者可以参阅 INEX 的网站，很多报告发表于 SIGIR 论坛[4]。现在已经有一些 XML 检索技术的应用 [1261]，例如，图书检索从 2007 年开始作为 INEX 的一个任务进行研究 [884]。 584

13.8 文献讨论

许多本章中讨论的方法的在 Ozcu 和 Liu 编辑的《Encyclopedia of Database Systems》（数据库系统百科全书）[1236] 中的一些条目中有更详细的描述。关于 XML 检索，这些条目包括 Kamps 写的 "Indexing Units" [874] （参见 13.4.2 节）、Kekalaineen 等人写的 "Contextualization" [893]、Trotman 写的 "Processingstructural constraints" [1597]、Pinel-Sauvagnat 写的 "Propagation-based structured textretrieval" [1264]、Tsikrika 写的 "Aggregation-based Structured Text Retrieval" （参见 13.4.3 节）和 Ramirez 写的 "Processing overlaps" [1332]（参见 13.4.4 节）。

我们对文本检索模型结构化能力的分析是基于百科全书中由 Hiemstra 和 Baeza-Yates 所写的 "Structured Text Retrieval Models" 条目 [761]，它本身是基于 Baeza-Yates 和 Navarro的综述 [115]，综述中还包含了对结构化文本检索的其他一些复杂模型的介绍。MacLeod 所写的综述也受到大家关注（尽管较旧一些） [1070]。Burkowski[298, 299] 提出了一种非覆盖区域的模型。Clarke 等人 [392] 扩展了这个模型，使之具有覆盖能力。基于邻近结点的模型是 Navarro 和 Baeza-Yates[1178, 1179] 提出的。在文章 [1071] 中，MacLeod 提出了一种模型，它基于结合属性和层次结点的单一层次结构（对于数据库类的查询）以及两个结点间的超链接。Kilpelainen 和 Mannila[906] 讨论了通过指定部分模式从层次化文本中检索的问题。在文章 [415] 中，Consens 和 Milo 讨论了查询文本区域的代数方法。

在结构化文本检索的先驱论文中，对 INEX 影响最大的是 Chiaramella 等人所做的 FERMI 项目中的一部分的工作 [371]。虽然只有理论分析，但是提出的框架是相关性定义的基础和 INEX 中研究的许多检索任务。

从 2002 年开始，XML 检索工作开始在 INEX 框架下评价。许多方法发表在每年在德国 Dagstuhl 召开的 INEX 研讨会上 [602, 606, 609, 610, 611, 612, 621]。这些论文集的链

---

[1] http://www.w3.org/TR/2007/WD-xpath-full-text-10-requirements。

[2] http://www.w3.org/TR/2007/WD-xpath-full-text-10-use-cases。

[3] http://www.w3.org/TR/xpath-full-text-10/。

[4] http://www.sigir.org/forum/index.html。

接可以从 INEX 网站上获得[1]。一些早期的 INEX 方法发表在由 Fuhr 和 Lalmas 编辑的 INEX杂志专刊上 [607]。后面的关于 XML 检索的研究发表在 Baeza-Yates 等人编辑的关于 XML 检索的专刊上 [119]。我们作为例子在 13.4.3 节介绍的语言模型框架在 XML 上的应用，是基于由阿姆斯特丹大学开发的 XML 检索系统 [1476]。

在 INEX 举办之前，SIGIR 会议[2] 主办了两次关于面向内容的 XML 检索的研讨会 [103, 334]。Luk 等人给出了一个早期的关于 XML 检索的综述 [1065]，它覆盖了很多关于索引和搜索
585 结构化文本的方法和挑战。近年来 Amer-Yahia 和 Lalmas 给出了概述（尽管短）[47]。关于这个概述的更多内容可以参考 Lalmas 所著的书 [965]。

虽然没有在这里进行讨论，但在数据库搜索中，提出了大量的方法来进行结构约束的模糊匹配（例如 [45, 1438]）。例如，[43] 提出的框架通过使用简单的查询松弛引入近似结果，从而扩展查询结果。然而，大多数方法并没有对它们的有效性进行评价。

INEX 数据集的介绍和活动可以在百科全书中由 Kazai[883] 写的 "INitiative for the Evaluation of XML retrieval (INEX)" 条目中找到。Lalmas 和 Tombros 详细地按照年份介绍了 INEX 的评价方法 [968]。XML 检索早期的测试集是 Shakespeare 测试集[887]，它影响了 INEX 的建立。对 XML 检索性能的评价本身就是一个研究问题，因为在评价检索性能中考虑结构会引入一些问题 [631, 886]。因此，组织了关于 XML 检索评价的研讨会 [1598, 1600]，对评价方法做出重大决策。其中一个具体的研究问题就是开发合适的、用于 XML 检索有效性评价的测度。大量的测度也被提出，其中一些在百科全书中由 Pehcevski 和 Piwowarski 所写 "Evaluation metrics for structured text retrieval" 的条目中进行了介绍 [1252]。

XML 检索中一个重要的研究领域就是所谓的整合信息检索和数据库 [44]，其目的是为面向内容和面向数据的 XML 检索提供方法。近期的在关于 "Ranking XML query" 的研讨会中交换了关于将其引入到 XML 检索中的最新的结果和挑战 [46]。Blanken 等人编辑了关于这两类方法的一部论文集 [210]。最后，Schenkel 和 Theobald 在百科全书中所写的 "Integrated IR and DB" 条目 [1437] 提供了关于这个研究方向在 XML 检索中的介绍，特别是 INEX。

Delgado 和 Baeza-Yates 写了关于面向内容的 XML 检索的 XML 查询语言的综述 [488]。NEXI 的介绍及其历史可以在百科全书中由 Trotman 所写的 "Narrowed Extended XPath I (NEXI)" 条目中找到 [1596]。全文 XQuery 的介绍基于 [42]。

致谢

本章是与《Encyclopedia of Database Systems》（数据库系统百科全书）[1236] 在结构化文本检索领域的相关条目一起写的。这些条目影响了本章的内容，反之亦然。我们还要对
586 Benjamin Piwowarski 的反馈表示感谢。

[1] http://www.inex.otago.ac.nz/。

[2] http://www.sigir.org/。

多媒体信息检索

——由 Dulce Ponceleón 和 Malcolm Slaney 著

14.1 介绍

14.1.1 什么是多媒体

在 Web 时代，我们生活在不断增长的海量数字化数据中，它们通过有线、卫星、音乐和视频下载、个人数码相机等到达我们的家中。这个趋势随着包含摄像头的手机的出现，而进一步加快，人们可以广泛地拍摄数码照片和短片，并上传到 Web 服务器。快节奏的数字化数据出版和分享，促进了娱乐、通信和计算机等行业的融合，也进一步巩固了数字化数据出版和分享的趋势。

由于大量可用数据的存在，很多新应用中都提出了提高检索和操作数字化数据的需求，例如视频点播、IP 语音、医学影像、多媒体数字图书馆、监控、安全、远程教学和事件检测等。其基本含义是，我们要为如下定义的多媒体开发更好的管理方法、过程和工具：

> *多媒体实质上是包括纯文本在内的任何数字化数据，这些数据大多数是非结构化的，用于通信或捕获信息。除了文本外，多媒体还包括视觉和声音数据、图片、图形、图像、视频、动画、语音、音乐、声音，甚至 3D 可视化。*

除了数据外，多媒体还包含复杂的空间和时间关系，这是从捕获、编辑、压缩和处理数字媒体的生命周期中派生出来的。在每一个阶段，核心的挑战之一就是在多媒体数据中发现隐藏的结构。 587

14.1.2 多媒体检索

在最普遍的形式下，多媒体检索问题可以表述如下：

> *多媒体检索系统的任务是检索与用户兴趣相关的文字、图像、视频和声音数据，并按照它们与用户查询的相关程度排序。需要计算相似度（即排序）以增加用户找到相关答案的可能性。*

对于检索，用户可以通过描述视频中的一个场景，例如“在电影《The Matrix》中，Keanu Reeves 在直升机坠毁中避开子弹”。为了能够处理这种需求，我们可以人工对每个电影中包含的数以千计的场景进行标注。虽然花费很大，但是这个工作只需要做一次。较少花费且具有较高价值的是对教育录像、监控录像、各种网络媒体中发现的经过很少量处理的视频片段进行自动地索引。这些索引可以用来回答用户给出的查询。这也是多媒体检索中最新的工作，例如，基于已标注的影片检索相片、图片或场景。

2000 年左右，网站开始准许外部用户来贡献他们生成的内容。这个趋势逐渐加速，因此，目前所有人可以很容易地将他们的照片放在类似 Flickr 的网站上，在 YouTube™ 上发布视频，甚至在社交网络网站（如 Match.com™、Friendster™、MySpace™、Orkut™、Twitter™ 和 Facebook™）上介绍他们自己。多媒体检索感兴趣的是，这些网站全都准

许用户来对内容进行标记。然而，这些标记却是非常个性化的，对检索和排序来说都是挑战。

多媒体信息检索（Multimedia Information Retrieval，MMIR）对多媒体数据中的信息和语义进行搜索、索引、提取、浏览和摘要。它包含不同的子领域，例如：

- 内容表示和多媒体对象表示，例如在图像和视频中提取底层特征（颜色、形状、纹理）。
- 特征提取。
- 查询表示，将高层语义概念映射为底层特征。
- 示例查询。
- 相关反馈，交互查询。
- 快速特征索引和编目。
- 集成搜索和浏览。
- 588 基于内容的多媒体搜索技术。

本章对多媒体信息检索的所有方面进行概括几乎是不可能的。这里，我们选择了多媒体信息检索中几个有代表性的领域。我们从对比多媒体和文本的不同，以及讨论处理多媒体的挑战来开始介绍。

14.1.3 文本检索与多媒体检索的对比

文本检索在很多方面与图像、音频和视频检索不同。在文本中，可以自然地把单词当做基本单元，标点和段落提供了结构信息。即使是字体特征也显示了强调。相反，多媒体数据是典型的连续流，是包含很少分界符的线性故事。对于非文本媒体，语义单元的定义是取得高质量搜索的基本步骤。例如，在视频中，时间是很重要的，内容随时间变化，因此需要利用视频分割将连续的流分割成可以管理的块，我们将在14.7节中进行讨论。

由于语音识别的发展，现在可以生成具有很低错误率的高质量语音文字转录。然而，即使是接近于完美的转录也缺乏标点、段落这些提供结构的元素。因此，虽然基于语音转录的检索似乎非常接近于文本检索，但是实际中却不是这样。处理这个问题需要认识到，在语音转录中单词所附着的时间信息对于检索是十分有价值的。例如，单词间大的空隙可能会指示话题的转换。

文本文档和多媒体对象大小上的绝对差异也构成问题。包含100页内容的文档通常需要200KB的存储空间，但是一个75分钟的音频，在压缩为MP3文件之后，仍需要60MB的存储空间。而且，存储1小时的MPEG-1压缩视频需要600MB，为了没有延迟地传输，则需要1.5Mbit/s的带宽。

在多媒体中浏览也不同。尽管现在的图形显示令人印象深刻，但是传统上我们有很强大的专注于文本的技术文化，对于单词有良好的定义和相对良好的文化理解。比如，摘要和加亮对于文本来说就比较容易理解。对于多媒体而言，没有权威的或者一致认可的方法来定义哪些素材应该出现在文摘中，也没有一致认同的展示给用户提供最好的方法。

与文本检索相比，多媒体检索是一个比较新的学科。虽然如此，图像和视频检索引擎的增长，以及刺激人们管理和分享照片和音乐的网站及应用，都预示着多媒体已经扎下根来。

本章大致按照从上到下的方式组织。我们先介绍基本的多媒体检索，之后我们再详细介绍机器学习技术。图14-1给出了在多媒体检索系统中信息流的几个阶段。

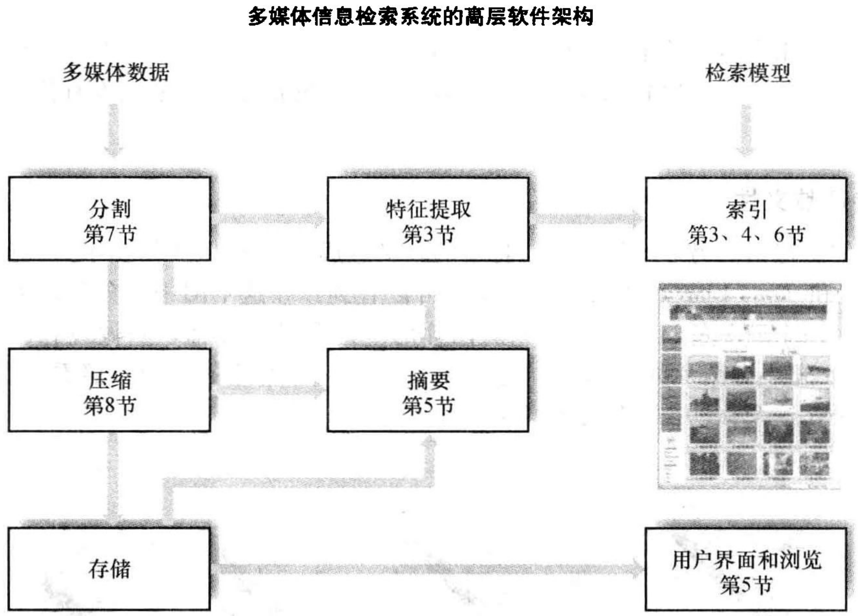


图 14-1 多媒体信息检索系统的高层软件架构，展示了多个阶段的信息流

14.2 挑战

14.2.1 语义鸿沟

在多媒体信号的内容和意义之间通常存在着很大的鸿沟。这通常称为语义鸿沟，图 14-2给出了象征性的说明。工资表中数字和它的意义之间的语义鸿沟非常小，文档中的单词及其总体信息和含义间会有一些语义鸿沟。而视频和它的语义之间有着较大的语义鸿沟。 589

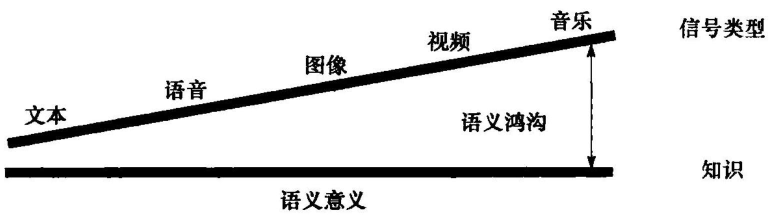


图 14-2 语义鸿沟，即多媒体信号与其意义间的鸿沟，越大表示信号的类型越复杂，例如从单词到音乐

物体识别是图像和音频处理中最难的问题之一。然而我们全都可以做到：看到一个图片，即使不到 1 秒钟，我们也可以从中发现人脸和其他种类的物体。虽然在商业系统中光学文字识别和（也许）人脸识别可以工作得够好，但是它们不能直接应用于更一般的对象。标注图像成分或者在波形上分析声音都还是没有解决的问题，也是许多研究的主题 [148, 418]。今天的多媒体信息检索系统很依赖于用户生成的单词（或者标签）[516, 1644]，并且在生成答案时大多数忽略掉了内容特征。 590

即使在图片中物体是已知的，图像或者声音信号能够携带主观性和情绪解释等复杂的特性，但这些很难用计算机来重复。在语音中，这些非语义信息由信号的韵律传达。这就使得

“不要停”和“不要！停！”之间产生了区别。但是，即使在最好的情况下，这种情感信息也很难识别，更不用说传达给信息检索系统的用户［1492］。虽然有些语音回答系统确实在寻找韵律信息来对客户的情感状态进行评估，但绝大多数的信息检索系统没有任何方法来表达这种信息。

14.2.2 特征歧义性

多媒体信息检索的另一个挑战是由特征的歧义造成的。图 14-3 给出了窥孔问题的例子。根据在 $T=0$ 和 $T=1$ 时刻的两张快照，我们可以看到，在这个图中一条线向右移动。为了效率的原因，一个简单的运动检测器只对图片的一部分进行评估，叫做窥孔。窥孔将决策（和计算代价）缩小到图片的一部分，我们可以从图 14-3 右边的两个快照看到。不幸的是，当从窥孔看时，这个移动被看做是对角的。

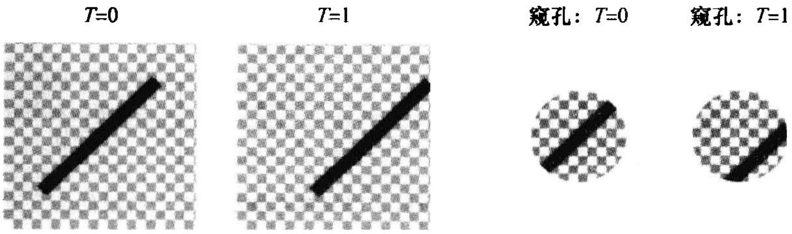


图 14-3　一个运动窥孔问题的例子。图中的线条是向右移动的。但是当从一个很小的孔来看（通常为了效率的原因会这样做），这条线看上去是向右下角移动

造成窥孔问题的原因是在一个窗口（窥孔）内缺乏全局信息导致的。当我们考虑整个图像时有很强的证据，特别是线的端点出现更复杂的变化。但是这种自顶向下的综合用基于机器的算法来实现也是困难的。

14.2.3 机器生成的数据

因为数据的不断增长，多媒体是挑战同时也是热点。图 14-4 估计了未来几年多媒

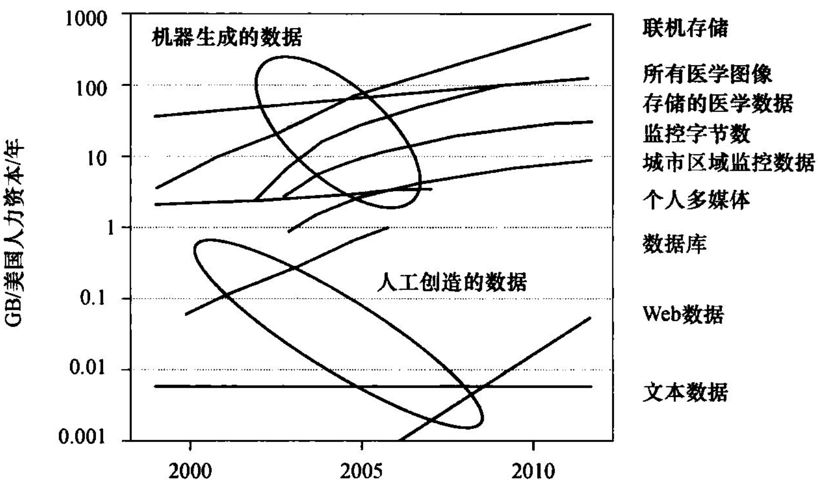


图 14-4　机器生成的数据随时间增长。需要注意的是，纵轴上的单位是 GB/人/年。感谢 IBM Almaden 研究中心提供图片

体产品的增长，以及能收集到的数据。这里显示的趋势是大概的。对于文本的估计是基于以下的假设来确定的上限：每个人一天打字 8 小时，每分钟 50 个单词，每天都这样工作（这个表格是平的，没有变化的，存在上限）。监控数据是由 2002 年在伦敦市中心固定摄像头的数量估计的，还根据这个推测了美国所有城市中摄像头的数量。医疗数据根据在全数字化医院中每个病人的平均字节数，并假设未来几年所有的医院记录都会进行数字化改造。人们制造并且消费了大量的多媒体数据，这需要大量的复杂的服务器群来正确地存储和检索。 591

14.3 基于内容的图像检索

在多媒体上的一些早期工作目标是发现和提取与图像内容相关的特征。虽然其原型产品在商业上并不成功，但是它们所开拓的技术是现在多媒体处理的重要组成部分。这些技术被概括地称为基于内容的图像检索。

问题

> 基于内容的图像检索的任务是基于图像内容进行检索。例如，考虑示例查询（Query-By-Example，QBE）方法。在这个方法中，用户提供一张图片，系统找到与其相似的其他图片。QBE 系统忽略掉图片所附着的语义信息，而是使用图片的简单特征，例如颜色、纹理、形状和显著点来检索和排序结果。

在所有的 QBE 系统中，最好的排序函数都是基于特征不变性的，即那些不会受到姿势、相机焦距和焦点、光照、相机角度和运动等变量影响的图像属性。但是这些变量的改变都会影响到在图像中组成物体的像素，这也说明直接使用像素进行比较是永远行不通的。代替像素比较的通常做法是采用整个图片的特征摘要，例如平均颜色。 592

14.3.1 基于颜色的检索

图像检索通常将颜色当做全局特征。这就是说，它并不依赖于图像的分辨率，即使颜色的位置对于目标感知非常相关。例如，考虑含有美国国旗及红、蓝条纹背景的美国邮票。两张看上去十分不同的邮票，也会具有几乎相同的全局颜色分布。

我们在 QBE 系统中使用颜色作为特征，比较不同图片间的颜色直方图。颜色量化到 N 个槽中的一个（我们会在 14.8.2 节讨论颜色表示），对每个槽中像素的数量进行比较。颜色直方图的好处就是，即使是视角或者构图的大改变也不会改变每种颜色的平均比例。也就是说，颜色直方图测度与视角和图像的分辨率无关。因此，不需要对前景和背景进行分割。在图像 I 中，颜色 c_j 的直方图定义为：

$$h_I(c_i) = P(color(p) = c_i \mid p \in I) \tag{14-1}$$

其中 $P(color(p)=c_i \mid p \in I)$ 表示从图像 I 中随机选取一个像素 p，其颜色是 c_i 的概率。

我们可以通过加入每种颜色在图片上的相关位置信息，即颜色的空间相关性［787］来扩充颜色直方图。为此，我们通过统计颜色对（c_i，c_j）的数量，以及颜色为 c_i 和 c_j 的两个像素之间的距离 r，构建颜色自相关图。表示为如下形式

$$h_I(c_i, c_j, r) = P(color(p_1) = c_i \wedge color(p_2) = c_j \mid r = d(p_1 - p_2)) \tag{14-2}$$

其中 $d(p_1 - p_2)$ 表示两个从图像 I 中随机选择的像素 p_1 和 p_2 间的距离。图 14-5 说明了具有相同颜色直方图的图片间自相关图的巨大不同［787］。

例子 1：基于颜色的内容检索

利用颜色直方图（或者自相关图）的图像检索通过预先计算（和存储）每个图像的直方

图来实现。给定查询图像，我们可以简单地查找具有最接近直方图的图片。使用这种技术的一个应用例子是 QBIC 系统所做的工作 [568]，如图 14-6 所示。

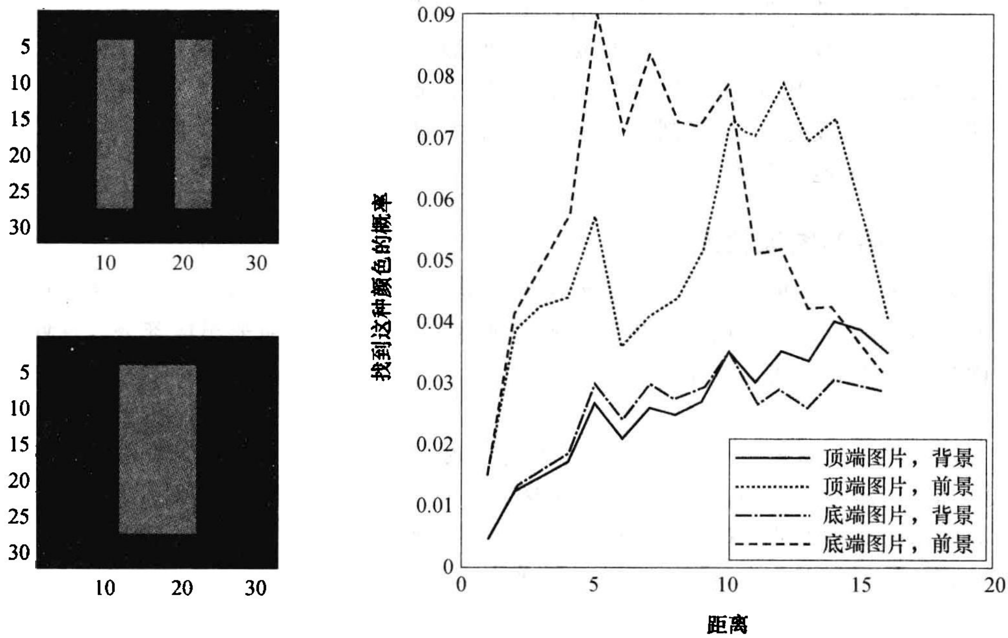


图 14-5 具有相同全局颜色分布（即颜色直方图）的两张不同图片的自相关图对比。注意，自相关图在前景颜色（黑）间的不同，即右图上面两条线

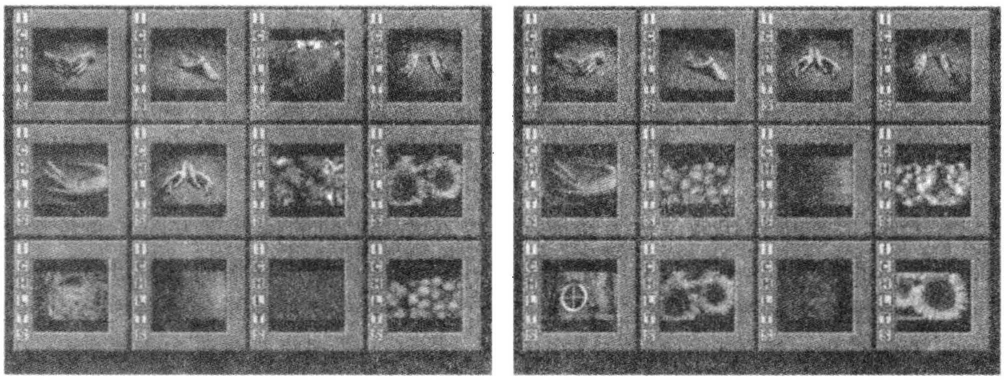


图 14-6 使用自相关图像检索的两个例子。左边一组使用平均颜色，而右边一组使用颜色自相关图。查询图片在每组图片的左上角。感谢 IBM Almaden 研究中心，QBIC 系统的 Jim Hafner

基于颜色直方图检索中最大的问题是感知特性，叫做“颜色一致性”。人类具有强大的识别物体颜色的能力，几乎不受光照的影响。苹果看上去是红色的，无论我们是在白天还是在室内灯光下看它。人类具有很好的感知同种颜色的能力，不受周边环境的影响，但是颜色直方图就不一样了。此外，物体的确切颜色对于它的识别的重要性很小。

14.3.2 纹理

在基于内容的检索中，第二个有用的特征是纹理。当我们想到纹理时，我们会想到在触

摸物体、织物、物质，或者任何能触知的表面时的感觉。例如，我们说“这个桌子的纹理是粗糙的、沙质的或者光滑的”。纹理也是图像中物体的一个属性——橙子图像的纹理和草地图像的纹理不同。与颜色类似，纹理也是图像和视频检索中的一个关键特征。然而，与颜色不同，它是图像的区域特征，而不是单点特征。 593

在图像处理中，纹理用来度量图像中元素的重复度。这是一个可感知的现象，很容易被人类发现，但是用数学描述却具有挑战性。更确切地说，纹理刻画了图像强度的重复模式，这些模式如果用来区分物体就太细微了。大多数的纹理度量与强度和方向无关。 594

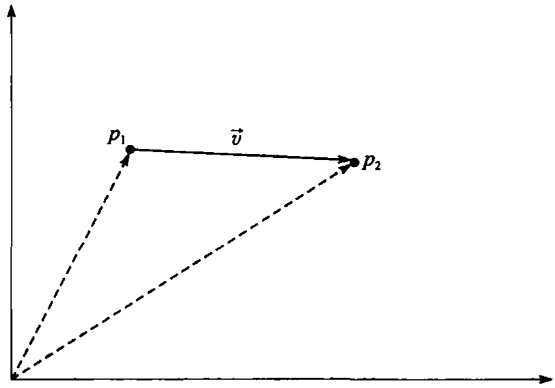


图 14-7　给定一个向量 $\vec{v}$，符合 $\vec{p}_1-\vec{p}_2=\vec{v}$ 的像素对 [p_1，p_2] 叫做 $\vec{v}$ 对齐。$\vec{v}$ 对齐的像素是 GLCM 矩阵计算中要考虑的

共生纹理测度

最简单的纹理测度是使用叫做灰度共生矩阵（Gray-Level Co-occurance Matrix，GLCM）[698] 的共生矩阵，它总结了图像中像素对之间光照模式的信息。要考虑的像素对由向量 $\vec{v}$ 确定，它确定了每个像素对之间的方向和距离，如图 14-7 所示。一个像素对（p_1，p_2）有由 $\vec{v}$ 确定的方向和距离，叫做 $\vec{v}$ 对齐。我们先定义第一个概率 $P_I(c_i, c_j, \vec{v})$，表示在图像 I 中找到 $\vec{v}$ 对齐的像素对，其颜色为 c_i 和 c_j 的概率。

$$P_I(c_i,c_j,\vec{v}) = P(color(p_1) = c_i,\ color(p_2) = c_j \mid \vec{p}_2 - \vec{p}_1 = \vec{v}_2) \tag{14-3}$$

其中 p_1 和 p_2 是图像 I 中的像素。给定了上述概率，不同种类的统计信息可以用来在 GLCM 中归纳信息，如下所述。

能量：关于 $\vec{v}$ 对齐的像素的亮度测度。

$$\varepsilon_I(c_i,c_j,\vec{v}) = \sum_i \sum_j P_I(c_i,c_j,\vec{v})^2 \tag{14-4}$$

熵：关于 $\vec{v}$ 对齐像素分布的非均匀性测度。

$$\Psi_I(c_i,c_j,\vec{v}) = \sum_i \sum_j P_I(c_i,c_j,\vec{v})\log P_I(c_i,c_j,\vec{v}) \tag{14-5}$$

其中 P、v、i、j 的定义与前相同。 595

对比度：关于 $\vec{v}$ 对齐像素对间的差异测度。

$$\mathcal{C}_I(c_i,c_j,\vec{v}) = \sum_i \sum_j (\phi_i - \phi_j)^2 P_I(c_i,c_j,\vec{v}) \tag{14-6}$$

其中 ϕ_i 表示像素的光照强度测度。

均匀性：关于像素间的相似度测度。

$$\mathcal{H}_I(c_i,c_j,\vec{v}) = \sum_i \sum_j \frac{P_I(c_i,c_j,\vec{v})}{1+|\phi_i - \phi_j|} \tag{14-7}$$

有兴趣的读者可以从纹理测度中找到更多信息，例如 14.10 节中的粗糙性和方向性。

例子 2：基于纹理的内容检索

图 14-8 给出了一个纹理检索的例子，由 QBIC 系统 [568] 实现。这个系统结合三种纹理测度——粗糙性、对比度和方向性，构造了一个特征向量。用户给定了一个样例图片，显示在每组图片的左上角，QBIC 系统基于纹理找到与它相似的图片。

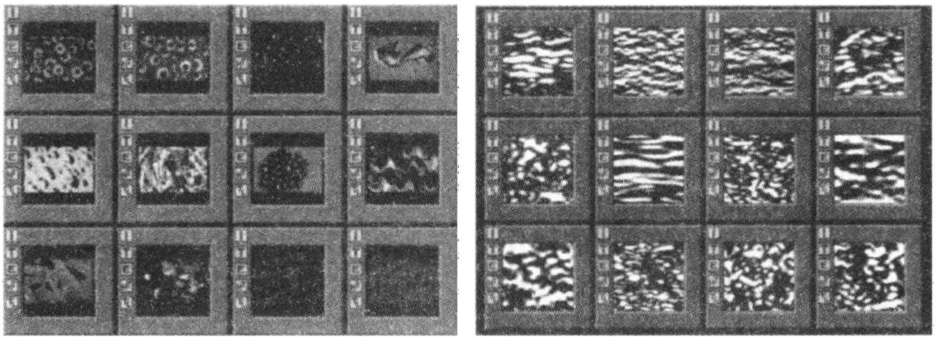


图 14-8 使用纹理的图像检索例子。每组中左上角图片是查询图片。感谢 QBIC 系统的 Jim Hafner[568]

14.3.3 显著点

基于颜色和纹理的检索算法都使用覆盖整个图像的全局直方图。更复杂的方法是构建一个特征模型，仅在图像“感兴趣”的区域结合颜色和空间频率信息。这个方法就是分析图像以找到其中特别独特的点（因此对于感知系统来说是显著的）。相同种类的显著点构成可计数的“单词”。

显著点［1051］是一种在图像中发现多尺度不变特征的技术。这些点对于包括光照、相机位置、相机或者物体角度在内的变换鲁棒性很好。显著点往往出现在角点或者图像中的独特位置，如图 14-9 所示。关于显著点的典型操作包括关键点［161］、稳态方向（stable orientation）和纹理中的局部几何［1525］。多种方法的比较可以参见［1612］。

596

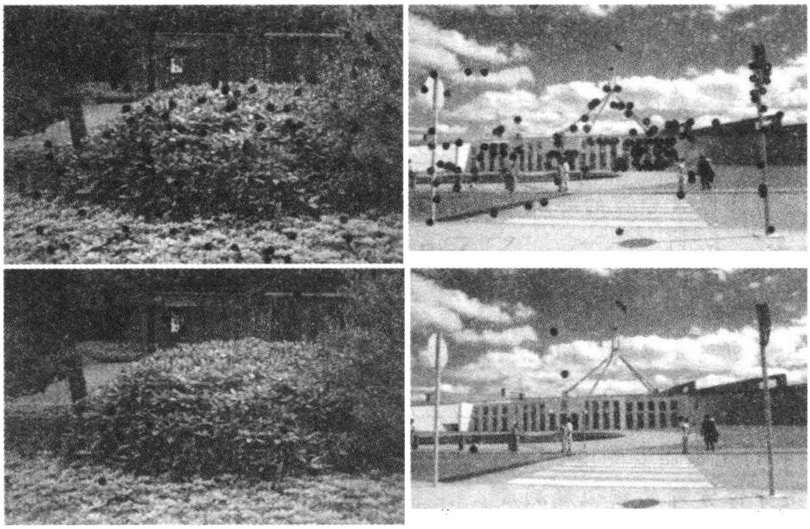


图 14-9 给出了在不同环境下两种不同种类的显著点。上面的图像给出了在叶子和建筑物场景中分别找出的典型的显著点，如例子 3 中所述。下面的图像给出了相同图像的另一种显著点。注意到只有很少的（建筑物）角点出现在叶子上，反之亦然。图片授权使用［1525］

例子 3：基于显著点的内容检索

图像相似度基于归纳图像中显著点的统计信息来计算。每个显著点周边图像的特征利用

简单的光谱过滤器进行刻画，它是基于人类感知的，对图像大小和方向变化抽样得到。这就形成了描述每个点的基本“字母”。将这些值进行聚类（如 k 均值算法）[517]，得到语言中的“单词”。图像用每个单词出现的次数来描述。类似概率潜在语义分析（probabilistic Latent Semantic Analysis，pLSA）的方法可以用做图像匹配 [242]。图 14-10 给出了基于显著点的内容检索的结果。

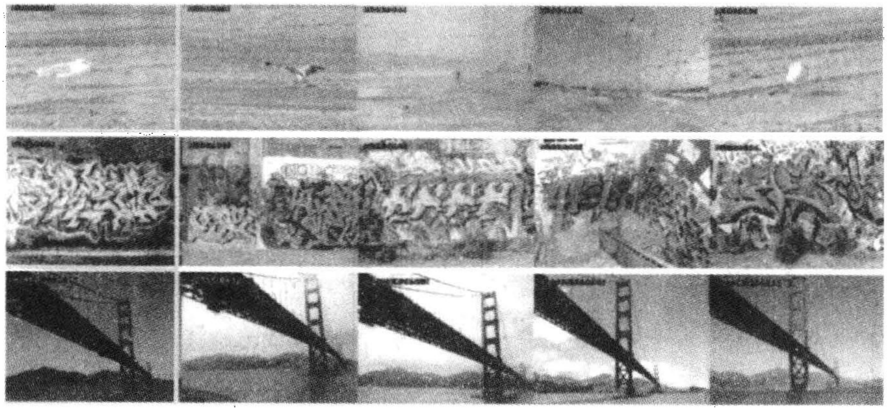


图 14-10 基于显著点的图像相似度计算任务的检索结果。每行中第一个图像是查询图像。图像授权使用 [776]

14.4 声音和音乐检索

在本节中，我们讨论多媒体中有关音轨和音乐检索的问题。这些问题有多种的形式，我们综合如下。 597

问题

声音和音乐检索中的基础问题是检索符合模糊定义的声音信息需求（查询）的音轨。这个问题有很多形式，例如：1）给定一个小的声音片段，找到一个与它相符的音频对象，叫做指纹识别（fingerprinting）；2）给定一个音轨，识别出其中包含的文本信息，叫做语音识别（speech recognition）；3）给定一个音轨，识别出里面的说话人，叫做说话人识别（speaker identification）；4）给定一个文本查询，检索符合这个查询的语音文档。

正如我们这里所讨论的，多媒体系统的任务就是解决这个问题中的各种形式。

14.4.1 指纹识别

音频指纹识别是音频信息检索任务中的一个成功商业应用。在指纹识别中，我们使用很小的一个声音片段来检索一个大规模的数据库，查找一个精确的匹配。这个过程很复杂，因为查询经常是损坏的——一个典型的例子是由移动电话在酒吧的吵闹环境下录制下来的一段声音片段。而信息检索系统的任务是从在 200 万（商业应用）首歌曲中找到包含这个查询的歌曲。这个任务叫做*指纹识别*，它变得越来越重要，因为我们希望收集很大规模的内容数据库，并在其中找到重复或者避免包含非法内容。

一个关键的方法是在声音的频谱时间分布（声谱图）上寻找大的变化，并且将这些声音信号中最显著的部分进行编码 [1664]。其难度在于这个过程在使声音信号发生劣化的常见情况下需要具有鲁棒性，例如，很大的背景噪声、移动电话上的廉价话筒和为了话音而不是

音乐优化的压缩算法。因为谱峰的位置相对稳定（即使加入了噪声），所以一系列的谱峰组
598 成了可用于识别的一段音频指纹，如图 14-11 所示。

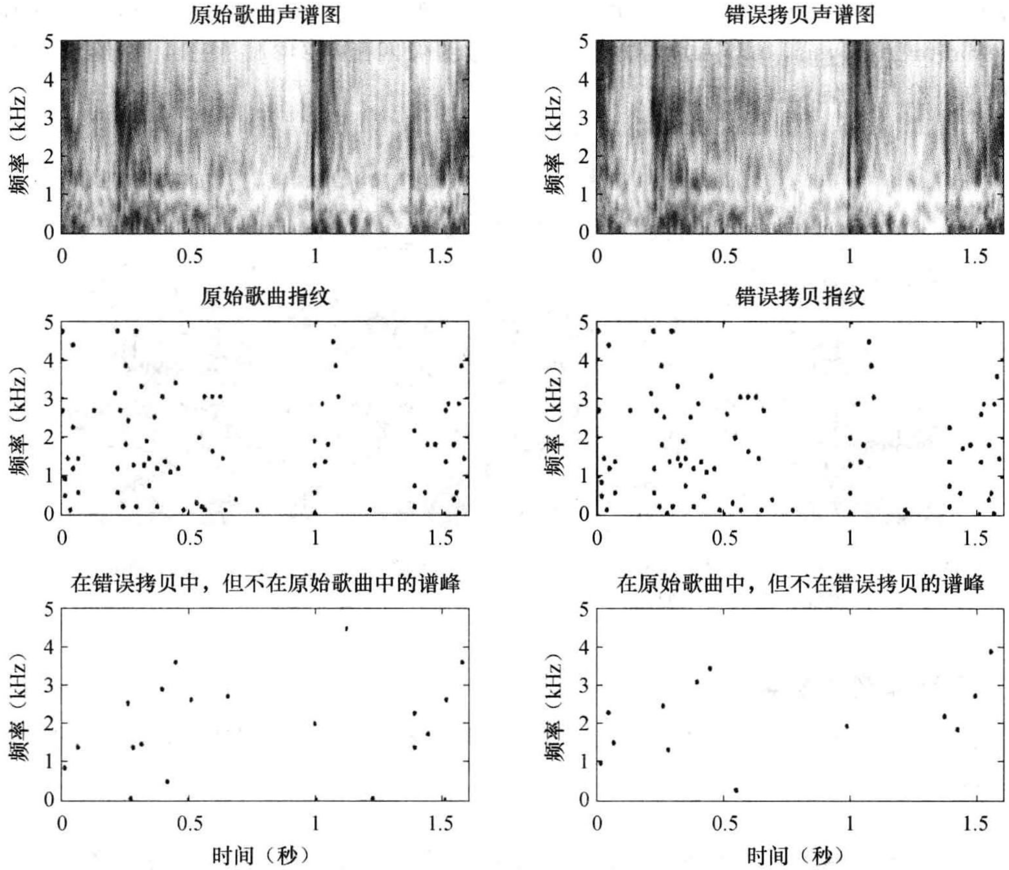


图 14-11 说明了使用 Madonna 的歌曲 "Borderline" 时，指纹识别的鲁棒性。最上面的图表示了原始歌曲的声谱图和一个带有错误的拷贝。中间图表示了谱峰的位置。在这个噪声层次上，原始歌曲和其拷贝间大约有 15%～20%的对应谱峰不相同

14.4.2 语音识别

语音识别是将包含在音轨中的单词识别出来的过程。如果满足两个条件，它可以达到较好的效果：1）听觉环境是受限的，因此麦克风只能听到单一的语音，并且没有背景噪声和音乐；2）任务是良好定义的，因此语言模型可以限定在任何一个时间点上需要识别的单词数量。但是，多媒体信号通常不能同时满足这两个条件，不论媒体是为娱乐开发的，或者是
599 在家里临时录制的，再或者是为了监控目的。

1. 隐马尔可夫模型

语音识别通常采用隐马尔可夫模型（Hidden Markov Model，HMM）来查找能最好地解释所录制数据的单词模型序列。这些模型包含合法的音素序列（语言模型，Language Model，LM）信息和它们的发音信息（音素的听觉模型，Acoustic Model，AM）。所有这些信息约束到单一的概率框架下，在听觉错误和语言错误中进行平衡。对于每组可能的音素，HMM 给出对应单词的音素序列与听到的声音间的概率估计。图 14-12 举例说明了关于这两个模型的一个简单的 HMM。

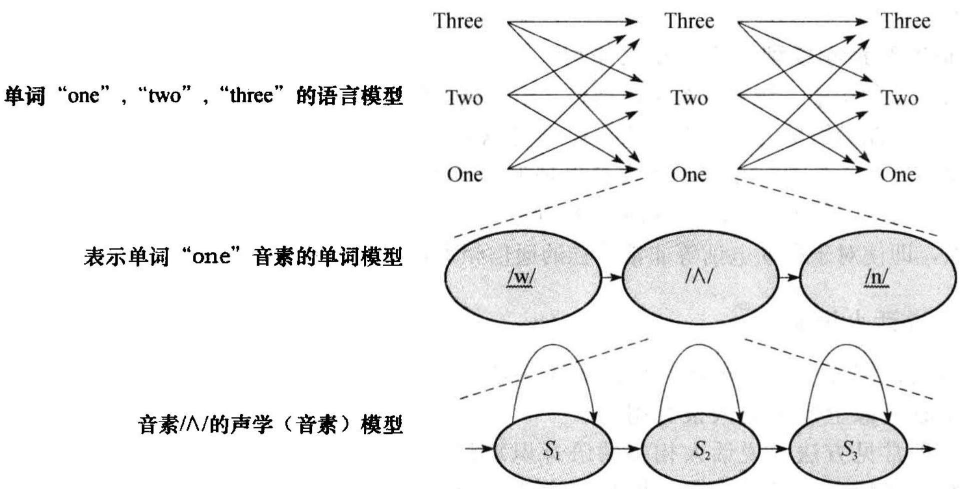


图 14-12 用来识别数字串的 HMM 模型的一部分。语言模型用来描述有效的单词，例子中包含三个数字。每个单词模型描述了用于对这个单词建模的音素。音素模型包含自环的三个状态序列，描述了音素怎么发音的。音素模型中的每个状态对应了特定的 Mel 频率倒谱系数（Mel-Frequency Cepstral Coefficient，MFCC）向量出现的概率

HMM 模型将语音信号建模为静止状态序列——认为信号是不变的，当它改变时 HMM 移向下一个新的状态。每个状态利用概率密度函数对语音信号的一部分进行建模。这个密度函数对说话人说出给定的音素时，在特征空间中每个点被听到的可能性进行建模。为了处理语音的动态性，每个听觉（音素）模型包含 3～5 个状态，每个状态使用高斯混合模型（Gaussian Mixture Model，GMM）描述 Mel 频率倒谱系数（Mel-Frequency Cepstral Coefficient，MFCC）向量。

2. 高斯混合模型

有多种方法发出单词 cat 中的音素/a/的音。在 HMM 中，这个问题利用每个音素的 GMM 模型来处理。GMM 模型是一个使用少量（混合）高斯函数建模的概率密度，它使用 600
39 维空间（因为 MFCC 向量有 13 个系数（14.4.5 节中介绍，我们增加了 13 个一阶时间导数和 13 个二阶时间导数）。多维高斯模型的基本形式是：

$$G(x,\mu,\boldsymbol{\Sigma})=\frac{1}{(2\pi)^{N/2}\,|\boldsymbol{\Sigma}|^{1/2}}\exp\left(-\frac{1}{2}(x-\mu)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(x-\mu)\right) \tag{14-8}$$

其中 x 是 N 维空间中的一个数据点，μ 是高斯均值的位置，$(\cdot)^{\mathrm{T}}$ 表示矩阵转置，$\boldsymbol{\Sigma}$ 是描述了数据间协方差的矩阵。通过叠加高斯模型来得到混合模型，每个分量表示听觉空间中不同部分的概率：

$$GMM(x,\{\mu\},\{\boldsymbol{\Sigma}\})=\sum_i A_i G_i\,(x,\mu_i,\boldsymbol{\Sigma}_i) \tag{14-9}$$

其中 G_i 表示一个如式（14-8）所述的多维高斯模型，A_i 表示权重系数。一般来说，由 MFCC 表示形式的对角协方差（否则需要太多数据），这些协方差矩阵都是对角阵。

在更复杂的情况下，/a/在“cat”中的音与/a/在“bat”中的音听起来完全不同。这是因为在一个音素前面和后面的声音会改变中间音素的发音。所有 ASR 系统都用这种上下文相关音素加以处理。如果有足够的数据，那么这种概率可以更准确地估计，并训练更详细的模型，从而为每一种音素出现的上下文构造不同的 HMM 模型。因此，对于在“cat”与“bat”的音素序列，就会有不同的三音素模型（一种与在其之前和之后出现的音素相关的音

素模型）。

3. 语言和声学模型之间的相互作用

语言模型使得语音识别系统可以运转。语言模型限定了可能的单词数量，大大地降低了出现错误的机会。在最简单的情况下，语言模型可以只准许 10 个数字出现。这种情况下，我们可以说词汇表的大小就是 10，语言的困惑度也是 10。典型的大词汇表语音识别系统的困惑度可以达到 60，表示一句话中每个给定的单词后面平均有 60 个可能的单词。因为识别器由语言模型高度限定，所以，即使对于移动电话等非常糟糕的通信频道，语音识别也可以生效。

14.4.3 说话人识别

说话人识别任务是确定谁在说话而不关心他们所说的话。这对于确定画面上的新闻主播或者在家庭录像上出现的人很有用。

有两种常见方法：说话人相关的语音识别和 GMM 密度估计。最好的解决方案是语音识别系统对每个特定的说话人进行调整。说话人相关的系统对每个说话人根据其发音特点进行了调整，从而有一个唯一的模型。但是，对于大规模人群收集说话人相关的信息是很花费时
601 间的，也不太现实。

替代的方法是用单一的（大型）GMM 等较通用的模型用来获取一个说话人的所有声音[1347]。这种情况下，GMM 可能会需要 2000 个分量来对说话人的说话方式建模。大量的分量是需要的，因为系统不是试图识别单一的单词，即所有的音素在任何时间都可能出现。使用 GMM 的说话人识别通常需要超过 10 秒钟的语音来做出可靠的判断。

说话人识别可以基于谁在讲话来帮助切分多媒体信号，我们将在 14.7.4 节中进行介绍。

14.4.4 语音文档检索

语音文档检索用来处理根据用户查询（可能是文本形式）来检索语音文档（spoken document）的问题。为了解决这个问题，最常使用的是两种语音特有的方法：关键词发现和音素识别。这两种技术都比一般的使用语音识别器进行言语转换的信息检索技术的鲁棒性要好。

第一个方法是关键词发现，即识别在语音文档中预先选择好的关键词。它是有效的，因为关键词通常在听觉上是很独特的。每个关键词包含很多信息，关键词的出现可以容易地发现，并且关键词具有很高的信息量。这个方法的应用是有限的，因为用户必须在查询中包含关键词。如果一个用户不能想起与信息需求相关的关键词，那么这个方法就不起作用[30]。

第二个方法是音素识别（phonetic recognition），即在音素层进行检索。举例说明，短语“it's hard to recognize speech”与“it's hard to wreck a nice beach”在语音上很相似，即使它们在字符串上不怎么匹配。因此，我们想独立于用户查询中的文本，识别出这些音素上的匹配。

音素识别的关键之一是处理在声音层的不匹配问题。使用传统的信息检索技术，单词“bat”和“bet”是完全不同的。但是，在语音上这两个单词上的/a/和/i/很容易混淆。Amir[48] 将这些类似的声音组合在一起叫做元音素（metaphone）。这些组包括元音，如 AA、AE、AH、AO、AW、AX、AXR、AY、EH、ER，以及辅音，如 TH、F 等。在一组内，元素认为是相同的。

14.4.5 音频基础知识

对于信息检索，音频有很多特性，正如我们在上面讨论的多个检索模型。最重要的是，

信号中的信息不能直接从音频的波形中使用——它像视频一样具有很大的语义鸿沟。因为在音频信号中的信息不是直接可见的，我们必须分析音频信号来从中提取基本信息。这是一个基本步骤，而且是音频检索系统中的一个重要部分。 602

音频使用波形来记录随着时间的推移气压沿着声波的变化。为了使这些记录具有原始保真度，并且不遗漏人可以感知到的任何信息，波形的测量必须多达每秒 44 100 次(44.1kHz)。包含一个很短句子的 1 秒钟的音频数据就有很多。

由多个声源混合产生的声波是复杂信号。例如，一个很大的湖及其周围。独立的声源包括湖上的船在航行，男孩子在扔石头，动物在涉水，以及鸟在觅食[⊖]。这些对象都在制造沿着水面传播的波纹。考虑连接湖的两个小通道，假设我们知道关于湖的信息都是通过进入这个通道的波形的叠加而推断的。解开这个混杂的成堆的信息是困难的问题，叫做听觉场景分析（Auditory Scene Analysis，ASA）。为了克服这个复杂性，我们假设每次只有一个声源，这就大大简化了从听觉信号中分析和提取信息的难度。

对于信息检索，每个声音场景中的对象有三个主要维度：响度、音调和音色。从信息检索的观点来看，我们可以忽略信号的整体响度——它包含了很少的信息。音调和音色包含了不同种类的信息。

音调是声音的一个属性，用来描述音乐的旋律。心理声学家基于我们的感知来定义它[1114]。语音研究人员根据咽喉中声门的动作来定义它。工程师按照信号的调和性来定义它。这里，我们使用音乐上的定义——我们最感兴趣的是演奏了什么音符。

> *音调是"声音的听觉属性，根据它，声音可以在一定尺度上从低到高排序"[778]。我们定义音调（或者音符）是在复合谐波上的最低频率。*

音调是听觉场景分析的一个重要提示，用来理解信号中的情绪内容，但是在语音处理中经常忽略它。

音色也是声音的另一个属性，用来发现正在演奏的乐器的种类。

> *音色是声音中的独立维度，我们将它定义为除了响度和音调之外的所有其他信息。*

为了理解信号中的情绪和音乐内容，我们根据组成声音的频率来查看音调。为了理解单词，我们查看音色。

1. 声谱图

声谱图（sound spectrogram）是声音的一种表示方式，用来描述信号的频率（频谱内容）随时间的变化。它类似于乐谱，但表示了不能放在乐谱上的更多信息。 603

图 14-13 给出了声谱图的例子。横轴表示以秒为单位的时间，纵轴表示频率，在图像上每个点的颜色深浅表示信号在当前时刻当前频率的能量。纯音音乐的频谱内容是乐谱信息的直接表示。在图 14-13b 中的其他线是基础频率的泛音或谐波。这些另外的泛音使得声音更丰富。图 14-13c 给出了钢琴录音的声谱图。每个音都伴随着很多泛音或谐波，丰富了声音。声音的音调还是保持一致的，但是音色改变了，因为钢琴有很多泛音。图 14-13d 是最真实的。这里演奏了长笛，并且音符的幅度也改变了，加入了颤音，周期性地改变了几个百分点的振幅，这也使得声音更加真实。音符在这里不是一成不变的，但是你还是会听到无歧义的曲调。声谱图展现了所有这些细节。

2. 音谱图

音乐信息检索系统依赖的声音表示称为音谱图（sound chromagram），即音度流[154]。 604

⊖ 感谢 Al Bregman 的比喻。

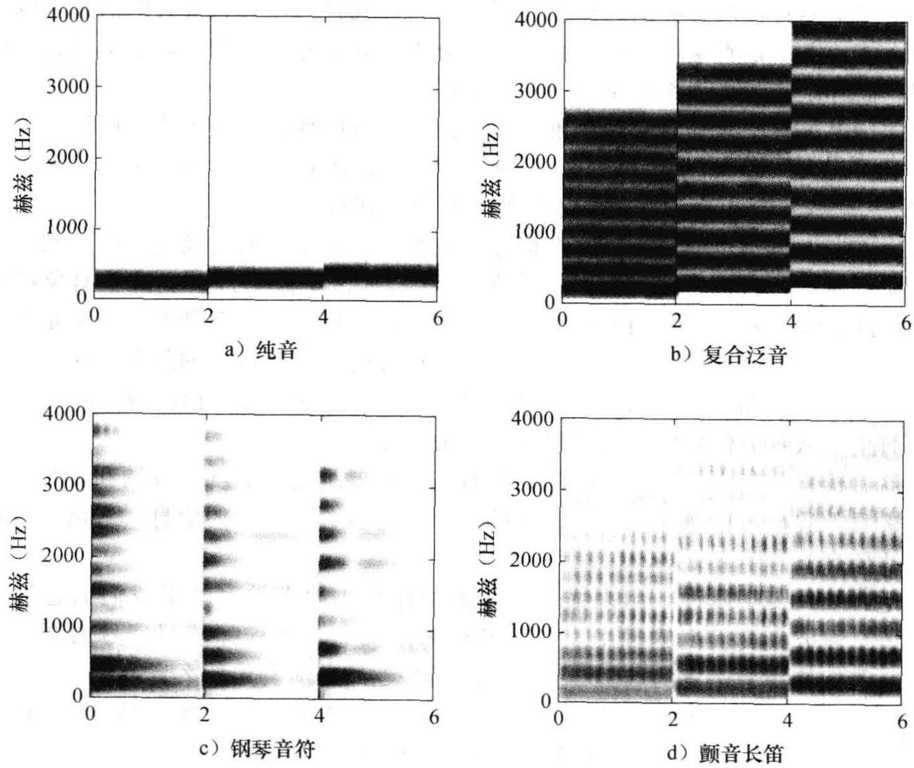


图 14-13 给出了对于 C、E 和 G 音符的四个不同的（窄带）声谱图：纯音、包含 10 个泛音的复合音符、钢琴音符，以及用颤音演奏的长笛。横轴代表时间。感谢 Kyogu Lee

音高描述了声音的音调随着其频率的增长而普遍增高。音度则完全不同，它是一个循环度量，对两个相隔八度的音调设置同样的值。我们说音度具有八度周期性。

音谱图是从声谱图中构建的，通过合并多个八度音节到一个 12 维向量。如果基础八度音阶使用 65～123Hz（C2～B3），那么信息从每一个八度音阶（频率从 65～131Hz、131～262Hz、262～423Hz 等）合并起来估计音谱图中的 12 个音符。这就提供一个八度音阶（即在钢琴键盘上的 12 个键）上的信息。在很小的波形时间窗口内，这个计算每秒完成 50～100 次。音谱图将音乐的音符（或音度）表示为时间的函数。图 14-14 给出了一个例子。当我们在歌曲中匹配旋律信息时，音谱图表示最有效。

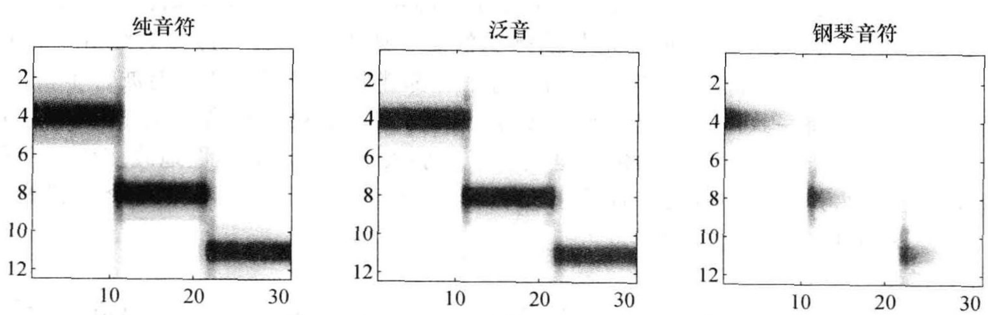
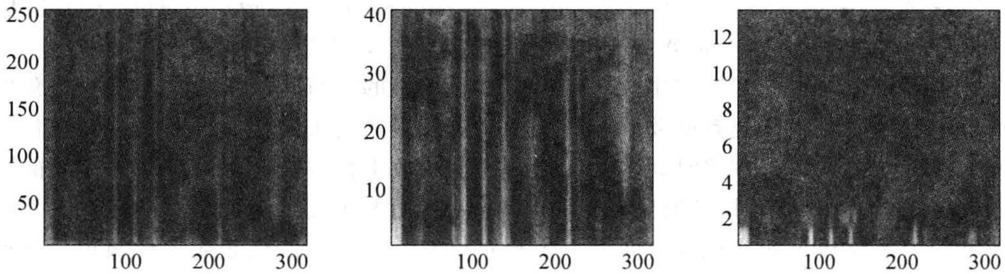


图 14-14 对于图 14-13 中所示的 3 个音符给出了 12 维音谱图，以时间为函数。感谢 Kyogu Lee

3. Mel 频率倒谱系数

语音信息是通过声音的音色表达的。音色最常用的表示形式是 Mel 频率倒谱系数（Mel-Frequency Cepstral Coefficient，MFCC）[476]。MFCC 将频谱的*主要形状*（broad shape）转换成为低维向量，如图 14-15 所示。MFCC 对频谱的每一帧进行操作，将详细的频谱信息转化为（通常）13 维的向量，用来捕获频谱图的主要形状。首先，对频谱信道进行重新采样和合并，用来模拟一个 40 维蜗形滤波器。MFCC 使用对数来模拟人耳对响度的感知——这是对响度的一种压缩。其次，使用离散余弦变换（Discrete-Cosine Transform，DCT）[212] 来降低维度。由于以下两个原因，使得最后一步很重要。首先，通过只保留 40 维 DCT 输出中的 13 位，使频谱信息得到了平滑，丢掉了包含在泛音中的音调信息，它们在声谱图上表示为一些细小的水平线。其次，DCT 具有一个有用的属性，即输出系数通常是不相关的。这就意味着可以使用简单的概率模型——概率分布的协方差基本是对角化的，所以它可以用对角协方差高斯密度来建模，大大简化了任何机器学习步骤。 605



a）声谱图　b）滤波器组　c）Mel倒谱系数

图 14-15　给出了计算语音信号“a huge tapestry hung in her hallway”的 MFCC 所需的处理步骤：a）声谱图；b）重新变换到 Mel 域滤波器组；c）最后，通过 DCT 将维度降低到 13 维

14.5　检索和浏览视频

科学文献中描述了多种多样的视频可视化、导航、浏览和摘要的方法。这些方法从简单的汇总，例如传统的故事板和动画，一直到能够翻看视频集的创新性交互式表示。在本节中，我们首先定义什么是视频汇总或者摘要，并讨论制作这种支持高效浏览的摘要所面临的挑战。其次，我们讨论三大类视频摘要：静止的（基于帧的）、动态的（基于视频的）和交互式的摘要。提供的示例显示了视频摘要和浏览技术的演进。最后，我们将对比视频和音频浏览。

14.5.1　视频摘要

视频摘要是一种视频表示方式，能够简洁而高效地表达视频内容。对于一个不熟悉这段视频的用户，摘要应当比原始视频更容易理解。也就是说，视频摘要是一种使视频浓缩表示或者可视化的技术。

视频摘要在提供上下文和覆盖率的前提下，应该简洁且一致。当视频摘要记录了原始视频的所有关键主题或者事件，我们说它*覆盖*（cover）了视频内容。虽然视频摘要可以人工生成，但是我们重点研究自动生成技术。

大家也许会想到类似 VCR 的方式（例如：快进和快退）可以用来快速查找和浏览视频。但是，这不是典型的案例。通常情况下，用户往往会超过或者未达到他感兴趣的场景 [67，230，383，533，1747，1561]。考虑到这些类似 VCR 方式的局限性，生成在上下文

中的摘要对于高效的导航和浏览是至关重要的。

视频摘要技术通常包含如下几个步骤：1）分析和分割原始视频为可管理的单元；2）利用从原始视频流中提取的视觉、音频和文本等特征的组合对这些单元排序；3）选择相关的
606 单元/片断来定义摘要；4）生成可视化摘要。

可视化模式可以分为两类：静止的（基于帧的）或者动态的（基于视频的）。研究人员使用视频摘要和缩略视频来表示静态摘要。动态摘要通过从原始视频中生成一个新的视频流来构造，通常是一段很短的视频。

14.5.2 静态摘要

我们先讨论静态显示视频摘要方法——静态意味着可以打印在纸上。最简单的视频摘要就是它的标题，即文本摘要。再复杂一点，可视化的摘要是基于一组静止的从原始视频中仔细挑选的图像（关键帧），有时还伴随着其他一些信息，例如字幕和时间戳。静止摘要提供了对于完全视频的一个紧凑的替代品，因为它们由静态图像组成。它们代表了超越类似 VCR 控制（通过加速内容）的第一步，不需要声音同步，也容易自动生成。

在电影制作中，编剧和导演用故事板来规划要拍摄的情节，故事板描述了相机角度，提供了整个电影的摘要。在视频摘要中，故事板对应于视频的关键帧，由一维（幻灯片）或者是二维（矩阵）缩略图组按照时间次序组成。早期的故事板方法比较简单，关键帧的选取或者是随机的 [1136]，或者按照时间顺序 [1560]。它们的主要缺点就是不能提供上下文，即很难判断一个故事板页面在视频时间线上的相对位置。对于长的视频或者视频集，早期的故事板一定不能作为导航工具，因为它们需要上下滚动和切换页面。

更智能地抽取关键帧的方法是基于镜头或者场景。一个镜头可以用一个或者多个关键帧表示。颜色、纹理、运动等底层特征的组合都可以用来选取关键帧。尽管它们有缺点，但静态故事板还是广泛应用于视频检索系统和类似 iMovie 的商业产品中。例如，图 14-16 给出了将时间信息引入幻灯片的可视化表示，其中缩略图与一个立方体结合，它的深度按比例反映了镜头的持续时间。

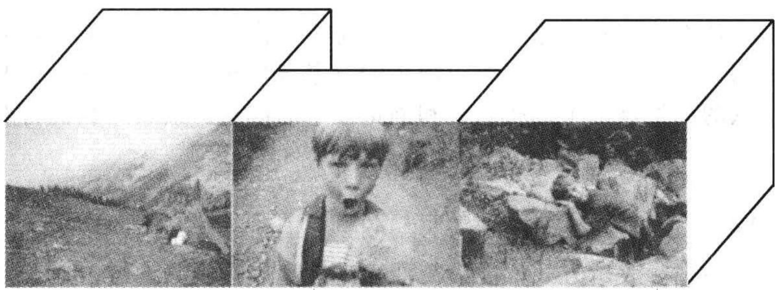


图 14-16 给出了一个 ButterBarSkim 的例子，它的缩略图与立方体结合。越深的立方体代表越长的镜头。许可后使用 [1497]

复杂故事板

传统故事板中的缩略图大小相同。二维故事板提供了另外一种方式，其中缩略图有不同的大小，其出发点是相对大小可以表示关键帧的重要性。视频漫画（Video Manga）（图 14-17）受到单词 Manga（即日文单词“漫画书”）的启发，代表了这类故事板 [1613]。在漫画中，不同大小的缩略图由一个类似于漫画书风格的、赏心悦目的形式组成。挑战在于需要高效地

对不同大小的缩略图进行排版，使得它们可以填满空间，还可以表达视频中的时间顺序。一个特殊的帧背包算法可以用在这个方面，详细介绍参见 Vchihashi [1613]。交互式版本的视频漫画准许用户浏览一组视频。 607

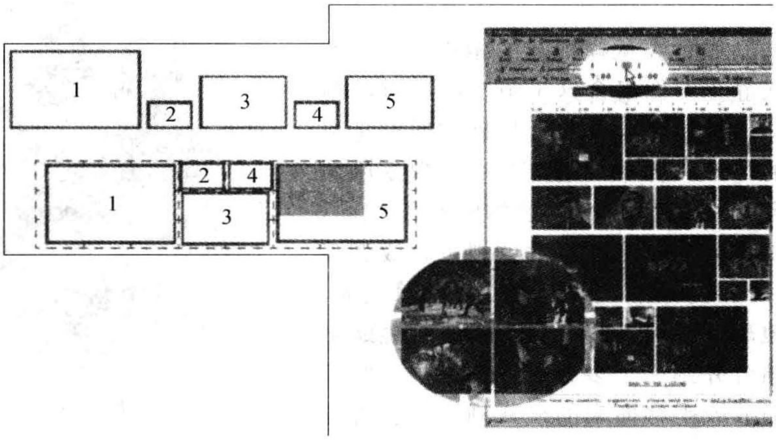


图 14-17　漫画是视频的画报式摘要，根据日文单词漫画书命名，其中缩略图的大小反映了其关联的关键帧的重要程度。经许可后改编 [1613]

14.5.3　图像拼接与跳跃剧照

对于包含移动物体和相机运动的镜头，例如倾斜和平移、缩放和改变焦距，单个关键帧不能有效地表示潜在的动态内容。在此背景下，研究人员使用更复杂的方式，利用图像拼接或跳跃剧照（salient stills）合成全景图片来表示一个镜头。 608

跳跃剧照是一类混合图像，由一个镜头或视频序列中的时间改变合成。根据运动是通过摄像机还是对象引入，跳跃剧照可分为三种类型 [1573]：平移、缩放和时间戳（Time-print）。图 14-18 给出了平移的跳跃剧照，图 14-19 给出了时间戳的例子。

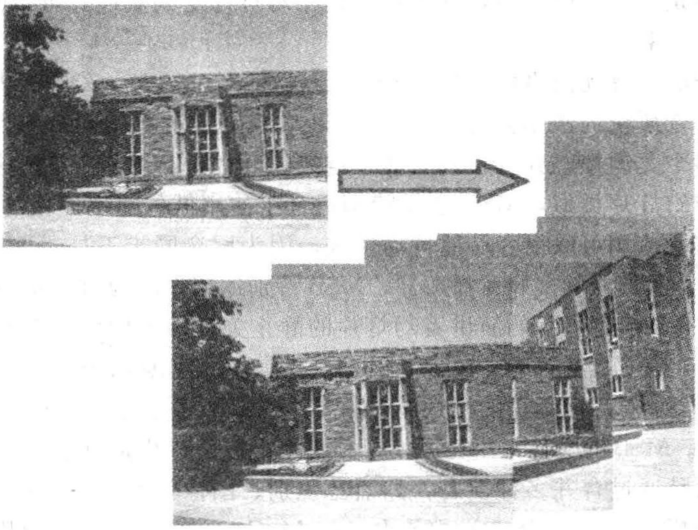


图 14-18　平移拼接：按照时间找到不同图像上的重叠部分，将它们合成为一个新的拼接图片。最后的处理步骤将图像合成为一个无缝的图像，这里没有显示。感谢 Sarnoff 公司的 Harpreet Sawhney

生成跳跃剧照需要两个主要步骤：建模和渲染。建模过程估计两帧之间的一致性程度。渲染过程选择基准帧和要渲染的帧，以及如何处理与背景相关的对象以及需要运用什么种类的时间操作。

平移的跳跃剧照计算从帧到帧的相机运动，产生单一的全景式静止图像组合这个镜头中所有的帧[1428]。相机运动必须计算得非常精确，通常是离线方式来进行的 [1100]。需要注意的是，跳跃剧照图像看起来与组合中任何一个单独的帧都不相同，例如会有更高的全局分辨率、更大的视野，或者多分辨率补丁。一旦跳跃剧照对所有镜头的计算完成，用户可以快速地理解视频内容。对于缩放的跳跃剧照将多个关键帧合并到一个多分辨率图像中。时间戳也是一种从缩放或平移而来的跳跃剧照，它将场景中的对象合并起来，构造一个背景和物体位置的聚合体。

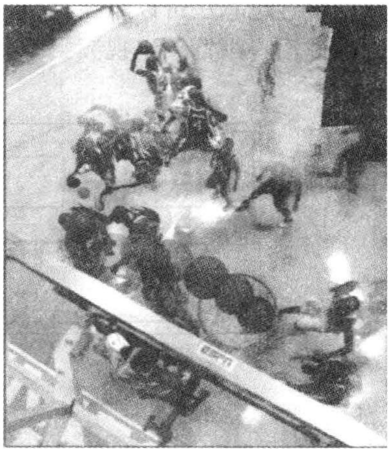


图 14-19 时间戳拼接：将一个视频的多个帧合并成一个图像来显示运动。授权后翻印 [1573]

14.5.4 动态摘要

一般来说，静态摘要不适合绝大多数信息包含在音轨中的视频，例如访谈、电视电话会议、技术讲座、教学和培训。动态摘要纳入时间和音频，同时提供紧凑性和非静态可视化表示。接下来，我们讨论这类摘要的例子，如幻灯片、移动故事板和电影预告片。

幻灯片（Slide Shows）的引入是为了解决长视频的静态视频摘要的弊端。幻灯片以固定的速率显示关键帧（按照时间顺序），并且包含播放控制和时间条（与故事板相比的重要优点）。在快速方式下，它们允许快速浏览长视频，提供了一个比原来的视频短很多的真正摘要。当下载速率是个问题的时候，这个特征使得幻灯片受到欢迎。可以使用不同的算法来选择组成幻灯片的关键帧。

接下来更复杂的是移动故事板（Moving Storyboard，MSB），幻灯片与原始音轨的一个可能具有较低的声音质量的版本同步。MSB 可以与原始音轨有相同持续时间。它抽取每个镜头的一个或者多个关键帧，并且在镜头整个持续时间中进行显示。对于长的镜头最好抽取多于一个的帧。与幻灯片类似，MSB 也是特别适合于使用低位速率连接的应用。

MSB 对于大多数应用可以很容易地自动生成，因为唇音同步不是一个问题。对于允许用户手动进行到下一个关键帧的多媒体播放器，MSB 特别适用，因为每一步可以前进一个镜头。在教室场景中，以高分辨率显示的演讲者幻灯片的静止关键帧比低位速率的视频流要好得多。因此这个方法适用于讲演或者音频中包含了绝大多数信息、且其中的运动是不重要的视频。但是，这个方法不适合于对高速运动的视频进行摘要，例如网球比赛、汽车追逐，或者其他任何
609 与运动相关的视频。MSB 的文件大小比原始视频要小，但是持续时间是由音频决定的。

更高级的界面是通过合并多种模态，如语音识别、图像处理和自然语言理解来自动处理视频。电影内容分析（Movie Content Analysis，MoCA）项目是最早使用多模态生成电影预告片（Movie Trailers）的系统之一。电影预告片是长视频的短版本，试图吸引观看者的兴趣。MoCA 定义了三个处理过程来产生视频摘要 [1030]。首先将视频切割为镜头，并识别出人脸、对话和字幕上的文字信息。其次，选择最能代表这个电影的片段。这里使用的方

法是专注于特殊事件（如爆炸）、主要演员、对话和字幕文本。最后，将这些片段按照一定顺序进行组合，并选取合适的过渡。作者描述他们结果的质量与手工编辑的摘要“相似”。然而，故事片似乎不太合适使用自动摘要，因为自动方法没有考虑故事的情感内容。 610

14.5.5 交互式摘要

最早的视频浏览界面之一是 Apple 视频放大镜（video magnifier）[1136]。它提供了整个电影的层次化浏览。从一行粗略的关键帧开始，每个关键帧可以扩展为另外一行，提供更详细的信息。这个方法很重要，因为它可以让用户在观看视频中的特定信息时，还可以保持对整个电影总体结构的认识。图 14-20 给出了一个故事板的例子。

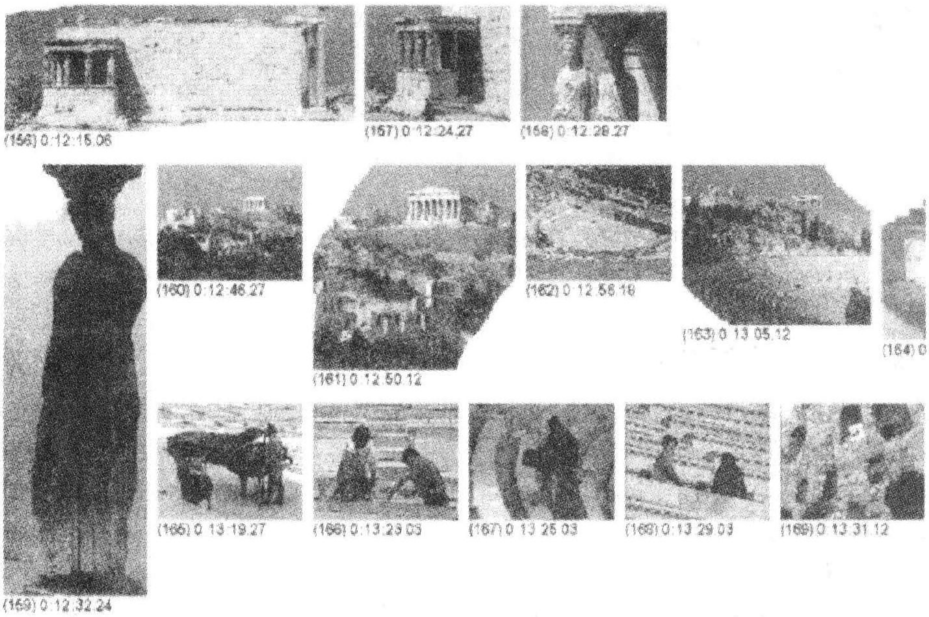


图 14-20 一个包含图像拼接和传统关键帧组合的故事板例子。PanoramaExcerpts 系统使用背包算法来优化排版。授权重印 [1561]

即使复杂的故事板也不能适用于视频和视频集，因为它们不具有简洁性和多功能性。一个解决方案是 MovieDNA，一种适用于视频、视频集，或者一般地，任意线性数据的可视化方法 [1289]。MovieDNA 要求将视频按照直接的方式（如基于时间）或者更复杂的基于内容的方式（如基于镜头）分割。MovieDNA 是一个二维图像，在图像上很像是一个 DNA 指纹。用时间（在一个或多个不同视频中）将图像串起来。图像中的每个像素表示特征出现在视频中的时间点。特征是一个人、主题、音频类型或其他元数据。用户可以很快地看到视频中有什么、什么时间出现，并且可以很快地跳到视频中合适的片段。将多个 MovieDNA 组合就变成了分层 MovieDNA（Hierarchical MovieDNA，HMDNA）。HMDNA 提供了一个视频集的高层概述。它是一个导航和可视化视频内容多层语义的有效工具，同时可以在视频集内保持位置和上下文。 611

图 14-21 给出了一个 3 小时视频集的两层 MovieDNA 的例子。在左边的第一层聚合了三个 MovieDNA，每个 1 小时。第二层在中间，表示了一个 60 分钟视频的 MovieDNA，每分钟 1 行，每一列表示一个特征。这个例子给出了视频集中第二个小时的数据，其中每一列中的特征表示从语音转录中自动提取的 *n*-grams。当用户在矩阵中移动光标（称为“刷”）的

时候，与这个部分相联的包含元数据的窗口会显示在右边，以显示第二小时视频的细节。MovieDNA 中第三个片段显示了一个作为代表的缩略图和与这个片段相关联的语音转录。

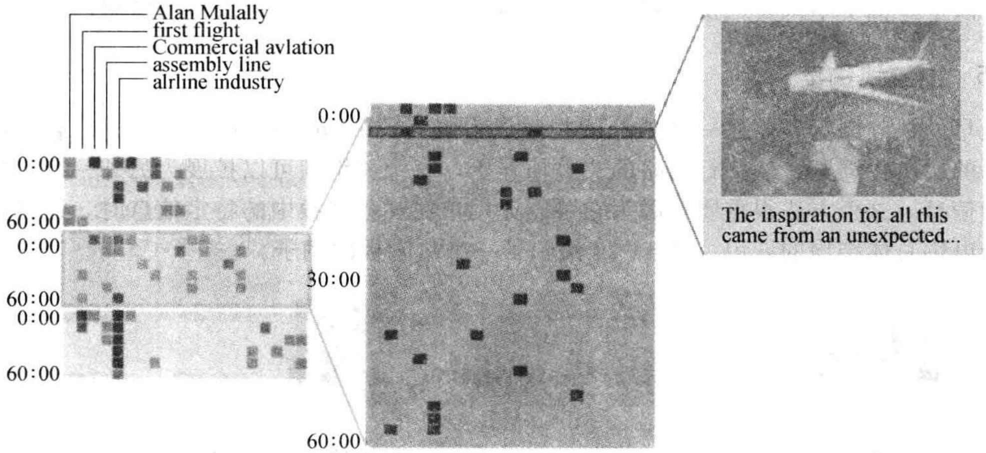


图 14-21 对一个 3 小时视频集的两层 MovieDNA 例子。来源于 Hierarchical brushing in a collection of video data，*Proceedings of Hawaii International Conference on Systems Science*（*HICSS*）（Ponceleón，D. B.，and Dieberger，A.），2001，© IEEE[1289]

14.5.6 视觉与听觉浏览对比

相比于浏览语音或声音内容，人们可以更迅速地浏览视觉信息。图像查询的结果可以在几秒钟内理解。类似地，一个中等大小的组里的图像内容可以很快理解。也可以通过并排方式比较相应的摘要来比较两个图像集 [1496]。对于特定类型的视频，我们可以利用快进方式来理解它们的视觉内容。

相反地，我们并不能迅速地浏览语音和声音。由于声音信号的时间属性，定义与缩略图单元相同的声音单元也是一个挑战。对于一般人，我们只能一次听一个音频流。从 1997 年
612 开始出现了能够提取语音片段的技术 [1435]，如去除静默、音乐以及其他非语音声音。

有两种加速音频的方法：时间尺度更改（Time-Scale Modification，TSM）和语音摘要（speech summarization）。TSM 算法产生压缩的声音信号，使得信号变短同时保持信号的音调、音色和话音质量。语音可以加速到 2.5 倍，普通用户仍然可以理解。进一步的提高可以通过修改较长的声音信号（如单词中的元音和语音中相对较轻的部分）来更快速地回放但仍保留原始声音信号中较短的部分（如辅音）。通过这种方法，语音可以加速到 4 倍，同时仍保证用户可以理解它的内容 [434]。当这些方法无效时，另外一种方法是分析单词，仅从中选取一些短语和句子进行回放。可以通过语音识别算法来提取文本和节奏，文本摘要算法用来选择最重要的短语，之后简单的音频编辑用来回放选择的语音 [1109]。

14.5.7 摘要评价

关于评价，并没有统一定义的评价指标来确定一个摘要的质量。在绝大多数情况下，评价是主观的，例如通过用户研究（user study）来决定用户是否可以成功地使用摘要而不是
613 原始视频完成特定的任务。关键是摘要质量的评价依赖于向用户提出的问题 [1564]。

自动生成的摘要很难证明是准确的，尤其是对于好莱坞影片。具有数百万预算的电影制作，可以很容易花费几万美金让人工编辑来制作预告片，可以从测试听众哪里得到最好的反

映。因此视频摘要可能最适合应用于体育赛事（对于不同的体育迷生成摘要）和家庭录像（只有很少的预算）。已经开发了一些特定的算法，尤其是对于体育视频，可以利用领域特有的属性。这些摘要使用欢呼声来检测特殊事件和得分，用 OCR 来检测比分，并提供著名运动员的信息，以及内置的关于角度、缩放和平移的知识。

14.6 融合模型：合并所有信息

多媒体融合指的是合并不同类型的数据，用于对多媒体检索任务做出更好的决策。这与文本检索不同，因为它经常需要从相同的多媒体信号中挖掘出不同种类的信息。

我们讨论两种不同种类的融合：基于另一个领域的信息识别本领域和同时使用两个领域来发掘感兴趣的信息。在第一种形式中，通过构建一个联合概率模型融合多媒体信号和文本模型来解决多媒体检索问题。这种类型的模型可以用声音来标注人脸、图像和声音。在第二种融合模型中，使用不同模态多媒体信号提供的不同种类信息，以便更好地了解这些信号。这种类型融合的最好例子是音-视频语音识别，通过视频上的唇读来提高语音识别准确率。

我们在多模态融合解决方案方面讨论四个重要问题：人脸命名、图像命名、音频命名和音-视频（Audio-Video，AV）语音识别。

14.6.1 人脸命名

Web 特别是新闻页面中包含了图片和它们的说明文字。这方面的内在问题是从说明文字中提取图片中人脸的名字。Berg 和她的同事用三个阶段来解决这个问题 [186]。他们首先使用基于主成分分析（Principle-Components Analysis，PCA）的标准技术来发现图像中的人脸。其次，他们使用简单的命名实体识别器来查找图像说明中合适的人名。此时，他们有很多面部图像可以用任何出现在说明文字的名字命名。最后，他们将标记了名字的所有面部图像聚类，来查找一组一致的图像 PCA 向量，能够聚合且在所有说明文字中最好地描述每个名字。我们先描述如何找到人脸。

人脸包含很多种类和姿势。然而，在大量的图像中，它们还是有共同的特征。Eigen-Face（特征脸）是识别人脸中公共特征的重要工具，它是通过使用主成分分析（PCA）查找最优子空间得到的 [1608]（参看图 14-22）。在特征脸中，所有的（训练）人脸图像都进行了对准，所以眼睛和其他脸部特征都始终在标准大小图像（$N\times M$ 像素）的同一位置。其次，从图像中读出亮度信息，通常按照字典序，组成一个大小是 $N\times M$ 的向量。每个人脸图像构成了非常高维的空间中的一个点。我们的任务就是将空间中对应人脸的部分从其他部分中区分出来。 614

命名实体识别器从每个图像关联的说明文字中抽取出常见的人名。通常，有些专有名词不会与人脸关联，如组织名，此外，还有一些图像中出现的人脸没有在说明文字中列出。最后的任务就是将图像和合适的人名进行对应。

对应问题通过聚类和期望最大化方法的组合解决。此时，我们有一批数据可以表示“George Bush”，因为它们都出现在包含“George Bush”说明文字的图像中，但是它们可能是其他人。Berg 构建了一个概率模型来分割特征脸空间。期望最大化（Expection-Maximization，EM）算法用来分配。首先，她估计出将特征脸空间与可能的名字联系起来的概率模型。其次，使用最大似然估计或平均估计，每个人脸图像分配了一个人名（或没有）。这个算法一直重复，直到名字-图像对收敛。Berg 在超过 1000 张图像的识别任务上，得到约 78％的准确率。

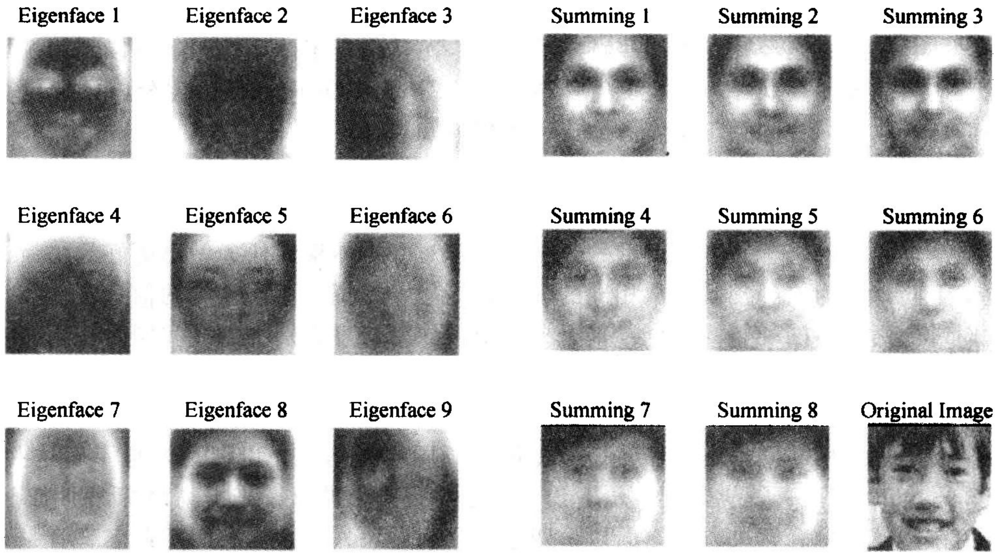


图 14-22 特征脸表示。左边的 9 个图像显示了通过分析 10 667 个 64×64 的注册人像图片，找到的前 9 个特征向量。右边的图片表示这些特征向量如何加起来逼近原始图像，原始图像显示在右下角

14.6.2 图像命名

一个融合图像和单词的一般方法是使用广义语言模型，在其中所有单词都用来描述图像的一部分。Barnard 提出将这个问题用机器翻译任务来解决 [148]。就像是从一种语言翻译为另外一种语言，他提出将图像特征与单词关联起来。通过使用层次图像聚类和接下来用一
615 组词来标记每个类来实现它。

分析图像的第一个任务是发现图像中对应于不同对象的不同部分——希望能够从天空背景中将老虎分割出来。一个方法是使用归一化割 [1463]。在归一化割中，构建了每个像素到每个像素的图。边的权重是两个像素间的相似程度的函数。对于更一般的情况，需要计算一个描述图像中较大区域的特征，以及两个像素在原始图像中的空间分离程度。想法是将结点（像素）分组，通过在图上选择好的分割，每个分割将完全不同的点区分开（因为它们的边的值高)。这个问题可以公式化为奇异值分解（Singular-Value Decomposition，SVD）问题。图 14-23 给出了一个样例结果。



图 14-23 给出一个利用归一化割进行图像分割的例子。从左到右：原始图片、利用 Canny 边缘检测器检测的边缘，以及利用归一化割找到的 5 个对象。感谢 Jianbo Shih 提供软件

图像中的每个对象根据归一化割建模为独立的区域。使用的特征向量包括大小、位置、颜色、方向能量和一些简单的形状特征。然后，使用这个特征向量将对象层次化分组，生成一个融合两种数据源的联合单词-图像概率模型。给定一个图像，我们可以查询单词-图像概率模型，估计单词最可能与哪个图像关联到一起。或者反过来说，我们要找到最能与单词对应的图像特征。

14.6.3　音频命名

Slaney 研究了类似的方式，但其目标是关联音频和单词 [1489]。这个问题在某种程度上比较简单，因为每个声音文件都假定只含有一个声音，因此不需要进行分割。但是，它也更复杂，因为声音表示是复杂的，声音可能长达数分钟且不断变换，而且它们的描述是句子的片段而不是关键词。在他们的系统中，两种不同的声效库中的声音通过文字描述信息链接起来（例如，马：一匹马在坑坑洼洼的小路上走近）。 616

锚空间（anchor space）将声音表示为点或者锚（anchor），对应于一个集成声音模型的距离。查询声音到每一个锚模型的距离组成一个向量。用 GMM 计算这些距离，就像是说话人识别中的说话人模型（参见 14.4.3 节）。

通过在两棵树上的结点间构建显式连接，将单词和声音关联起来。在声音层次簇中的一个结点与所有可能同这个声音相关的单词的多项模型关联。在语义空间中的一个结点连接到 GMM 模型所刻画的听觉空间的一部分（用锚模型距离向量表示）。图 14-24 给出了这些链接的数学模型。

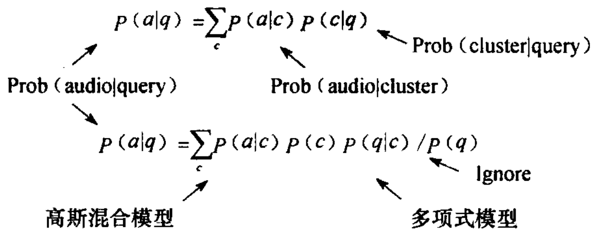


图 14-24　描述了在语义-音频检索中涉及的数学公式。第二个公式通过对第一个公式应用贝叶斯规则（Bayes Rule）得到

给定一个新的声音，我们查找它与每个锚模型的距离，找到在层次式音频模型中最符合的结点，然后从相关的多项模型中读出最可能的单词。或者给定一组单词，它构成了语义空间中的一个点，我们可以找到最相关的语义簇。然后，给定在锚空间中每个单词的位置，可以测试给定的声音和期望的音频空间部分的相似程度。

14.6.4　结合音频与视频的音-视频语音识别

音-视频语音识别（Audio-Visual Speech Recognition，AVSR）结合了与传统语音识别器类似的听觉信息和说话人的人脸视频信息，生成更准确的语音识别结果 [1298]。这是受到期待的，因为很多的声音，即使在很好的声学条件下，利用视觉信息也更容易辨认。在噪声环境下，听觉特征会产生误导，而视觉信息则完全不受影响，从而对语音识别提供了很强的语音线索，虽然此时的听觉场景几乎是没有用的。

我们用基于像素的，或者基于形状的特征来表示视觉证据。在基于像素的特征中，图像的像素通常经过特征脸（参见 14.6.1 节）等变化来构成特征向量。基于形状的特征表示知识的更高阶形式，其基础是查找面部特征的位置，例如嘴唇位置和下颌轮廓。两种形式的特征，或者它们的组合都可以用来作为 AVSR 系统的输入。 617

图 14-25 给出了两种解决 AVSR 问题的方法。在第一种情况中，称为早期融合（early fusion）或特征组合（feature combination），音频和视觉特征经过一定的归一化和重采样，合并为一个特征向量用于识别。在第二种情况中，称为后期融合（late fusion）或决策组合

(decision combination)，对两个特征流分别进行解释和分析，每个流提供一个关于单词出现的决策，之后合并形成最终估计。AVSR 使用的音频特征与传统 ASR 使用的相同，参见 14.4.2 节中的描述。

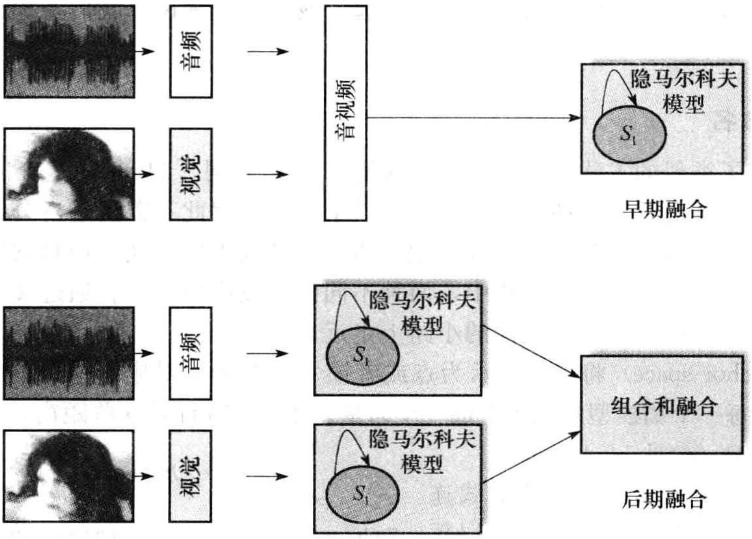


图 14-25 两种不同的音视频语音识别方法。顶部所示的是早期融合，音频和视频信息较早地融合为一个联合向量。底部所示的是后期融合，每个模态单独做出决定（分别使用隐马尔可夫模型），接下来进行融合

在最简单的形式中，听觉和视觉特征简单地连接，作为传统识别器的输入。一般来说，有很多特征操作阶段来选择最佳的维度 [517] 和旋转特征，以便让每个维度都是独立的 [82]。连接听觉和视觉特征称为早期融合，因为信息是在听觉降视觉做出决策之前的早期阶段合并的。

HMM 模型（参见 14.4.2 节）描述了由相对静止的状态序列组成的信号，每个状态产生一个特有输出特征集合。在听觉领域中，这些状态表示音素，对于英语传统语音识别可能会采用多种音素。视觉对应的是视位（viseme），一个特有的、用来表示嘴唇位置的视觉模式，大约有 13 种不同的视位描述出现在英语语音的不同视觉模式中。

618

在后期融合中，我们使用独立的识别器来对听觉和视觉信息做出决策，一个决策从听觉信息中识别音素，另一个从嘴唇数据中识别视位。然后，融合两个决策来决定哪个单词出现。

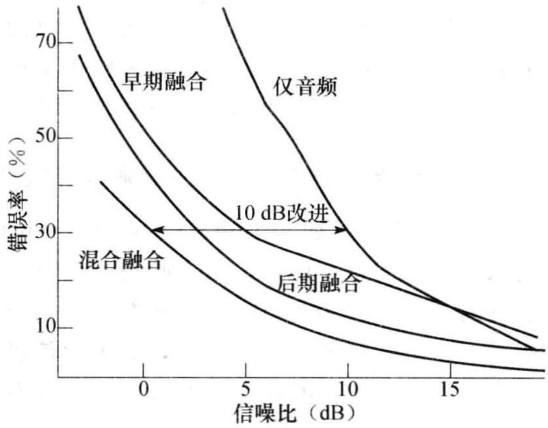


图 14-26 音频和音-视频语音识别的典型结果，显示了在使用视觉信息的情况下，容错度达到了 10dB 的信噪比（Signal-to-Noise Raion，SNR）。来源：Automatic recognition of audio-visual speech: recent progressand challenges, *Proceedings of the IEEE* (Potamianos, G., Neti, C., Gravier, G., Garg, A. and Senior, A. W.), 2003, © IEEE[1298]

图 14-26 给出了在一段范围的音频、视频和噪声条件（用信噪比衡量 [1298]）下

的典型结果。当只使用视觉信息的时候，单词错误率（Word-Error Rate，WER）相对高且不因噪声水平而变化。当没有噪声时，仅使用音频的结果是好的，但加入噪声后迅速下降。最后，AVSR 结果是最好的，特别是在高噪声的情况下。听觉信息和 AVSR 结果之间的差别，是加入视觉信号后的噪声容错度。在这个例子里，它是 10dB。

有趣的是，AVSR 使用后期融合比前期融合的结果好。似乎更符合逻辑的是，前期融合方法应该比后期融合要好，因为它拥有所有需要的、用于理解两个数据间关联关系和其他特点的信息。但是，实际上，在 AVSR 系统中早期融合不如后期融合的效果好。一种假设是，在进行早期融合时，联合概率模型太复杂，难以用单一识别器学习。这可能是因为模型不能捕获联合分布的细微差别，或者因为没有足够的数据。在两种情况下，后期融合，或者结合早期融合和后期融合的混合方式可以得到更好的结果。 619

14.6.5 结合音频和视频的多媒体处理

最后，更一般的音视频识别问题也可用上面介绍的 AVSR 技术来解决。

例如，IBM 的研究人员 [1171] 提出了一种多媒体系统用来标注视频不同部分的高层语义，例如“包含某人的视频片段”、室外场景、火、城市景观和飞机。他们测试了系统标注人物的精度，例如“Madeleine Albright”。最好的、仅使用音频的模型的精度可以达到 30%，而仅使用视觉信息的模型（人脸识别）的精度是 29%。非常吸引人的是，最好的、基于后期融合的综合系统的精度达到了 47%。这个数字表明听觉-视觉联合模型具有优势，也说明了仍需要做很多工作来提高整体精度。

14.7 分割

在实际处理用户多媒体查询之前，需要将多媒体对象分割成小的对象。这称为分割（segmentation）。我们讨论的分割问题的解决方法都是基于图像、音频和视频对象的基本特征，如颜色、纹理和光照强度。

当在信息检索系统中“加入”一个视频的时候，其中第一步是将视频分割为可管理的语义单元，即镜头，因为视频很大（通常，用 MPEG-2 压缩的 1 小时视频需要占用 2GB）。一个镜头对应于摄像机采集的一连串没有中断的帧序列。例如，采访类的视频会包含多种镜头，摄像机会在采访者和被采访者之间切换。视频由一些场景组成，一个场景定义为一连串相邻的、语义一致的镜头。相关的场景又合并成更高层的语义单元，在文献中有不同的名字，例如片段和故事 [414]。在电影或者视频中的单元层次结构如图 14-27 所示 [798]。

视频从一个镜头到下一个镜头间的变化有一个过渡（transition）。例如，突变（cut）是一个突然的过渡，因为整个图片瞬间变化，所以通常是一个容易检测的过渡。其他过渡，例如淡入（fade）、溶解（dissolve）和擦除则缓慢发生。信号中的噪声、相机闪光、特别的过渡效果，甚至是快速移动的物体都会使镜头边界检测具有挑战。这里考虑三种常见类型的场景分割：突变、淡入和溶解。

分割算法有多种类型，例如基于像素、差异性统计、基于颜色直方图、基于边缘、基于 DCT 和基于运动。这里讨论它们中的几种。我们还会讨论基于说话人识别的音频分割。然而，在我们继续阐述之前，我们先给出一个例子。

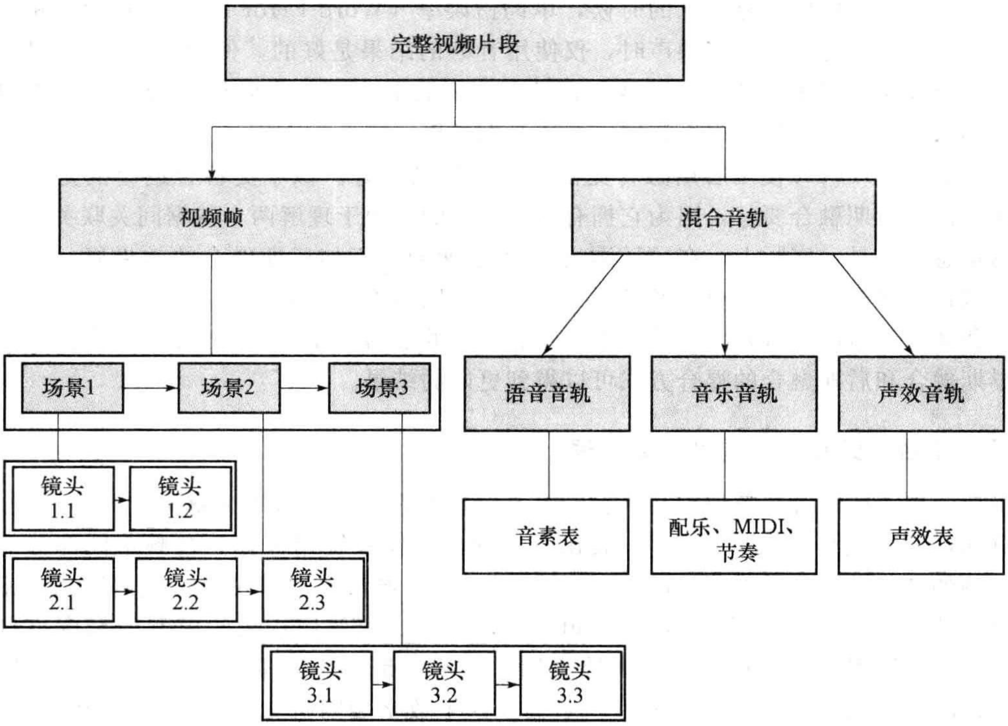


图 14-27 电影或视频中对象层级结构。根据 J. Hunter 的图片改造，2001 年

14.7.1 视频分割样例

图像的简单测度，例如两帧像素对之间的不同，不能对视频分割提供很好的信息。例如，当视频中出现常见的变化时，例如改变摄像机的拍摄角度，所有的像素都会改变，这样就会造成大的像素间误差。一种不同的方法是镜头边界检测，即通过在整个图像中查找统计信息摘要来发现在视频中哪些时刻发生了大的变化。基于统计差异性的早期方法将图像切分为不同的区域，随后比较这些区域的不同，例如两个区域间灰度级的均值和标准差。然而，
620 这些方法会产生很多误报（false positive）。

一个简单的全局统计值是图像的颜色直方图。直方图是通过图像中每种颜色的像素数计算得到的。因为颜色通常用 3 个 8 位的数字表示，所以颜色数太多了，很难直接用来计数。相反，我们将每维颜色大致量化到 8 个等级。这样一共有 512 种不同的颜色类型（8×8×8 种颜色）。直方图是镜头分割中最常用的方法。

图 14-28 给出了颜色直方图方法的例子 [1493]。这里，对于每一帧计算一个 512 个区间的颜色直方图。这样就会产生一个随时间推移的 512 维信号，每秒采样 30 次。奇异值分解（Singular-Value Decomposition，SVD）算法用来找到最佳的低秩近似信号。最重要的颜色信号（即 SVD 结果中包含最大能量的 4 维）作为时间的函数给出。这提供了视频中的全局变化的简单度量。如图 14-28 所示，我们注意到在镜头内部的信号变化比较小，例如帧中物体的运动（在 345 秒处）。但是镜头切换可以清楚地从颜色信号的巨大改变中看到，例如在 338 秒处。在 357 秒附近，两个相关的图像叠加用来提供视频片段间的平滑过渡，称为溶
621 解（dissolve）。

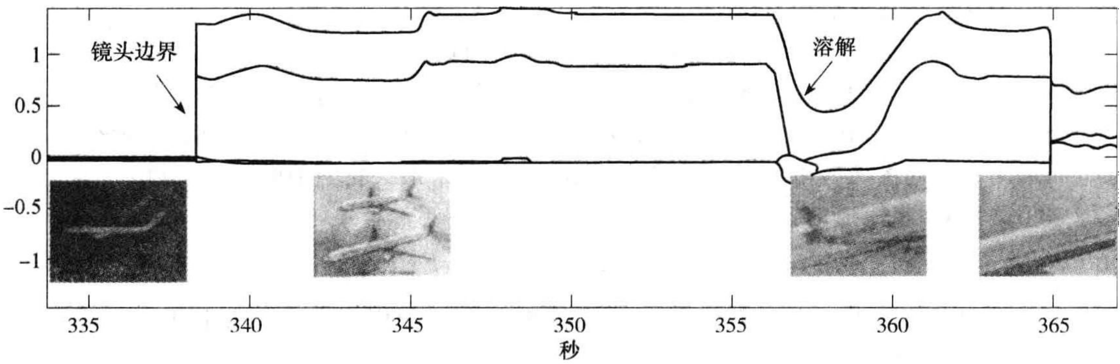


图 14-28　给出了一段 35 秒长的《21st Century Jet》视频片段的颜色信息摘要。信号中的急剧过渡对应镜头边界，但是大而慢的过渡表示的是溶解。在第 357 秒处，两种不同飞机的图像在溶解中叠加在一起。经授权使用 [1493]

14.7.2　视频分割方案

颜色直方图方法可以得到好的结果，因为视频变化是由于镜头或物体移动，或者灯光改变，引起颜色直方图缓慢变化——相比之下，大多数的图像不发生变化，只是重排整理。检测淡入更加困难，因为它表示了视频随时间的变化——开始时是正常的图像，视频线性地衰减为黑色，然后变到一个新的图像。一种简单的淡入检测器查找具有相同颜色的帧。亮度在变淡之前和之后符合线性曲线，可以通过计算帧平均亮度的一阶导数来检测。

溶解是最难检测的分割边界。在溶解中，图像通过交叉缓慢地溶解像素，从一个场景变化到另外一个，例如以像素到像素为基础在连续图像上执行线性插值。一种检测这种变化的方法是测量每帧图像的整体亮度方差。通常情况下，这一测度会相当高，因为之前和之后的图像包含我们可以看到的光照变化。但是在溶解过程中，两个图像会混合，而组合必然会降低整体的方差。因此，我们可以查找亮度均方差长达几秒钟的急降来检测溶解，如图 14-29 所示 [1029]。

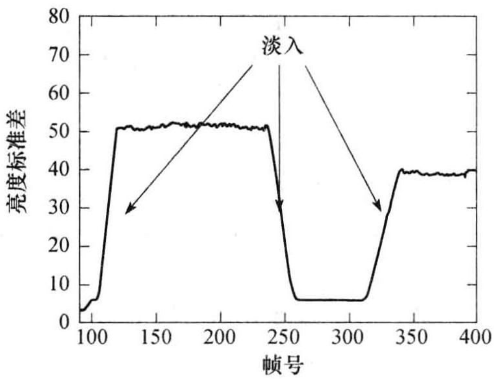


图 14-29　给出了三个淡入的图像在亮度标准差上的变化。经授权重印 [1028]

一种更鲁棒的查找溶解的方法是如 Covell 所做的构建显式模型 [433]。在她的综合分析方法中，她指出给定任意两个在溶解中间的点，都是端点的线性插值。这给了我们一个直接的手段来检测溶解。例如，我们可以在视频中以 1 秒钟为时间间隔采样帧对，并且查看中间的帧是否可以通过端点线性插值来预测。预测误差为我们提供了这一点是溶解的可能性的估计。我们可以扩展低预测错误的区域来找到溶解的开始和结束。 622

14.7.3　利用边缘的视频分割

更鲁棒的场景分割的测度是基于边缘的统计信息。边缘是图像中亮度急剧不连续的部分。它们对光照变化和镜头移动不敏感，因而受到关注。边缘的出现和消失告诉我们很多关

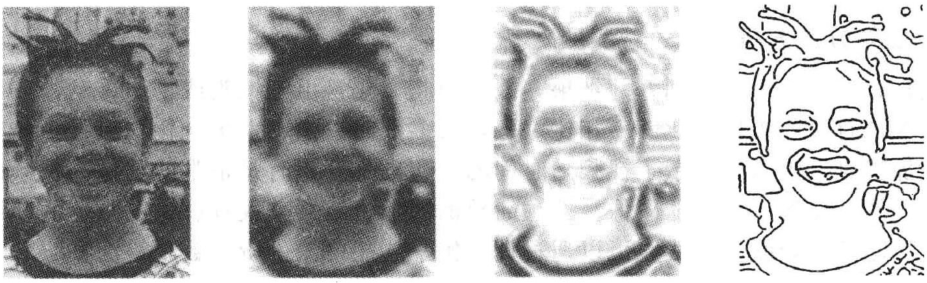
于视频中正在发生什么的信息。

Zabih 描述了结合运动估计和边缘检测，从而计算边缘变化率的系统［1762］。基本的思路是寻找不在序列的下一幅图像中出现的边缘（反之亦然）。这是有挑战性的，因为镜头运动会造成一帧中的边缘会出现在下一帧中的不同位置。因此，我们首先匹配两幅图像来消除全局运动。全局图像匹配是通过查找位移 Δx、Δy，来最大化当前帧（I_1）和下一帧（I_2）的像素相关性 C_{rr} 来完成的：

$$\sum_{x,y} C_{rr}(I_1[x+\Delta x, y+\Delta y], I_2[x,y]) \tag{14-10}$$

其中 x 和 y 是像素坐标。给定将两帧对齐需要的位移，我们现在有两张粗略对齐的图像，可以用来发现和匹配边缘。

Canny 边缘检测算子［327］找到图像中重要（对本算法的目的）的点。图 14-30 举例说明了这个过程。图像首先使用二维高斯函数 $G(x, y)$ 进行平滑，以模糊图像、去除噪声，并设置我们想看到的最小边缘。



a）原始图像　b）平滑图像　c）导数　d）边缘

图 14-30　利用 Canny 方法计算边缘的过程：a）原始图像；b）经过 6 像素标准差的高斯函数模糊后的图像；c）平滑后的图像的空间导数幅度；d）阈值截取后的边缘点位置

图 14-30b 给出了平滑后的图像。计算图像中每个点的空间导数幅度，可以得到图 14-30c。注意，急剧的过渡出现在黑色衣领的两边，在衣领的两侧有幅度很高的信号。最后利用自适应阈值在平滑图像中选择具有最大导数的点（图 14-30c），并确定边缘的位置。图 14-30d 给出了边缘的位置。

通过计算边缘在帧之间的往复来发现各种场景分割。对于每个边缘位置，在另一幅图像中的一小块区域中查找对应的边缘。在两副图像中都出现的边缘的数量提供了图像的相似度的度量。在静态或者缓慢移动的图像中，大多数边缘在一帧到下一帧之间不会移动太多。只有当整个场景变化时，由于突变、溶解（一切都改变）或者淡出（最终边缘会不可见），这个测度才给出比较低的相似度，因此是一个场景分割。基于边缘的检测对于运动和色度的变
623 化没有基于颜色直方图的检测那么敏感。

以上所描述的技术用于通过查看小范围视频来查找场景分割。然而，局部图像信息不会告诉我们帧如何组成场景或者更高层的故事元素。这需要通过层次分割方案来完成［1493］。

14.7.4 语音分割

分割边界通过判定信号中发生的改变来刻画。这个判定可应用于所有种类的信号（音频、视频和文本），但是我们通过音频来说明这个概念。进行这个判定的一种概率方法是通过构建信号第一部分的模型，将模型应用于接下来的信号，直到到达某个点，在该点模型不

再适用或不能解释数据，这就成为分割的边界。这种（单边）计算方法容易发生错误，因为即使一个新点可能不符合模型，我们仍不能确定这是由于噪声还是由于信号上发生了真正的变化而引起的。相反，我们可以使用双边方法，在一个可能的边界两边都比较信号模型。这个测试可以使用贝叶斯信息准则（Bayesian Information Criteria，BIC）来实现。音频可以表示为 100Hz 采样下的 13 维 MFCC 系数。

BIC 首先为较长的信号建立模型，然后再将信号细分为两个更小的部分，建立两个不同的模型。两个分开的模型总是会比一个模型更能拟合数据，因为有更多的参数来拟合数据。数据在边界的两端不同，因此两个不同的模型会得到最好的结果。当双模型比单模型更能预测数据，且获得的益处足以抵消额外的复杂性时，就可以进行分割。 624

当对比单模型方法和双模型方法时，BIC 检测通过引入对额外模型复杂性的惩罚项，解释额外的复杂性。给定一个模型 M_i 和数据 D_i，其中 $i=1$，…，N，BIC 计算 [1783]：

$$BIC(M_i) = \log P(D_1, D_2, \cdots, D_N \mid M_i) - \frac{1}{2d_i}\log N \qquad (14\text{-}11)$$

其中 d_i 是模型 M_i 中独立变量的数量。第一项表示这个模型以多高的概率解释数据的对数似然度。似然度越高越好。第二项对高复杂度的模型进行惩罚，因为它们使用更多的参数来描述。当使用 BIC 来预测分割边界时，双模型参数的数量 d_i 增加一倍，而数据点的对数变化量不大。因此，我们在信号中寻找某个位置，在这个位置，双模型以更高的似然度显著优于单模型，且其益处超过在公式 14-11 中额外参数的代价。

我们使用固定大小的窗口（长度大约 10 秒钟），在潜在边界的任一边，沿信号滑动边界来寻找使 BIC 测试在双模型上比使用整个 20 秒数据的单模型增益最大的点，这就是切分数据的最优点。

14.7.5 分割评价

镜头边界检测是相对成熟的研究领域。可以在文献中找到大量的方法，不只是检测，还有镜头和过渡效果的分类。多篇有代表性的综述文章比较了许多已提出的方法 [231，284，1029]。

早期镜头边界检测算法主要关注突然的过渡，即突变，随后渐变的过渡受到关注。已经提出了一些识别特定过渡类型的方法。然而，在实际中，这意味着通用的算法需要多遍处理视频。有些方法，例如那些使用全局统计信息的方法，需要通过人工或者自动的方法设定一个或一组阈值。实际上，只有自适应的阈值才有意义，因为找到一个适用于所有种类视频内容的全局阈值是不太可能的。开发不需要进行阈值调整，可以鲁棒地实时检测突变和渐变，且具有高的精度和召回率折中的单遍算法是一个挑战。

14.8 压缩和 MPEG 标准

与文本文档不同，我们很难见到没有压缩的多媒体对象——否则文件会太大。大多数的多媒体文件是有损方式存储的，基于特殊的算法将人脑不能察觉的冗余信息去除。例如，相比于感知颜色的变化，人的眼睛更容易感知强度的变化。因此，减少颜色变化信息，并不会影响人眼对图像的感知。严格地说，信息是损失了，但是压缩内容的*感知*（perceived）质量可以与原始内容的一样高。除了节省存储空间这一明显好处外，压缩可以使数字视频应用于对带宽要求很严的应用中，如视频点播（Video-On-Demand，VOD）和视频会议。 625

五个关键步骤使得图像和视频压缩高效、有用。这些因素是颜色子采样、利用离散余弦

变换去除空间冗余、熵编码、运动补偿和去除时间冗余。它们构成了典型的最新图像/视频压缩算法的基础步骤。这里不讨论 MP3 等音频压缩，原因有三个：1）解压缩音频不是很困难；2）（重新）计算需要的特征向量计算量不大；3）用于音频的有损和无损（lossless）原理与图像和视频的很相似 [830]。

14.8.1 强度和采样

颜色和强度是图片（或视频帧）中最基本的元素。因此，包括多媒体信息检索在内的图像应用构建在颜色和强度的数据之上。在照片成为多媒体信息检索系统的一部分时，已经可以通过图像传感设备采集图像。作为这个处理的一部分，光照强度作为时间和空间函数的离散点采样。如果图像采样太粗糙，那么信息会丢失；如果采样太精细，那么图像中会包含无用（冗余）信息。通常，我们可以放心地假设采样对图像内容和显示参数是正确的。然而，不论对什么颜色，强度信息通常都是均匀捕获的，我们可以看到，人眼对于不同的颜色和变换的敏感程度是不均匀的。这就提供了第一个压缩的机会。

14.8.2 颜色

颜色是图像的基本特征，人们可以感知和区分它。然而，人眼只对可见光敏感，可见光只是电磁波谱的很小一个区域。可见光波长范围是 400～700 纳米（nm）。每种颜色对应于此范围内很窄的频带，人眼可以区分 400 000 种颜色。波长大于 700nm 的构成红外线、FM 广播、TV 信号，而波长低于 400nm 的对应于紫外光和 X 射线。这些都不能被人眼感知。

人们通常使用对三种不同波段的颜色敏感的光感知器感知颜色。因此，所有颜色也通过红、绿和蓝（RGB）荧光体在图形显示器上显示。图像中的特定颜色通过特定强度的三种颜色产生，通常的范围是从 0（黑）～255（亮）。

由于颜色是通过 RGB 强度显示在屏幕上，这与人们视觉系统如何感知它们不同。因此，实际上有其他一些颜色系统，利用更接近人们感知颜色的方法来表示颜色信息。一种流行的颜色表示替代方案称为色调、饱和度和亮度值（HSV）。在这个方案中，基本颜色（红、绿、紫）使用色调的值来编码。亮度值（或光亮度）是整体的光源强度或能量。饱和度的数量表示颜色是品红还是深红——它们的纯度。相比于 RGB 等基于硬件的方案，这种颜色表
626 示是与感知相关的。

一个相关的颜色系统称为 YC_bC_r，用来作为图像（JPEG）和视频系统（MPEG 和 DVD）的基础。与 HSV 类似，YC_bC_r 系统利用三个值对颜色系统进行编码：亮度 Y、蓝色色度信号 C_b 和红色色度值 C_r。给定经过伽玛校正（对于强度值的一种非线性校正，可以使得感知强度更线性，从而弥补显示系统中的非线性特性）的 RGB 值，YC_bC_r 值可以通过如下三个公式给定：

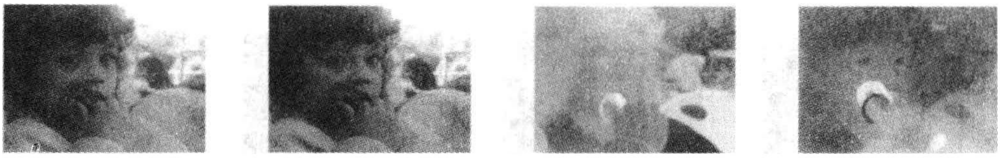
$$
\begin{aligned}
Y &= K_r \times R + (1 - K_r - K_b) \times G + K_b \times B \\
C_b &= \frac{1}{2} \times \frac{B - Y}{1 - K_b} \\
C_r &= \frac{1}{2} \times \frac{R - Y}{1 - K_r}
\end{aligned}
\tag{14-12}
$$

其中，R、G、B 的值表示在 RGB 方案中红、绿、蓝的强度，K_r 和 K_b 是常数，$K_r = 0.299$，$K_b = 0.114$。

降低颜色或者色度信息的采样率是图像压缩中的重要步骤。我们的眼睛对于在亮度（强

度）空间上的变化比在色度（颜色）上的变化要敏感。因此，在图像变换到 YC_bC_r 方案后，Y 信号保持不变，而 C_b 和 C_r 信号在水平和垂直方向的像素采样都会降低到原来的 1/2 或者 1/4。

图 14-31 给出了对于一个彩色图像降采样后的效果和它的三个分量。原始图像看上去很完美（即使在本书中显示为黑白图像）。但是靠近看，我们可以注意到 C_b 和 C_r 信号在水平和垂直方向上的采样率都降低 1/2。降采样在黑白图像中很清楚，但是我们在压缩的全彩色图像中却看不到。即使不考虑其他步骤（去除空间冗余和使用熵编码），仅降采样一项就可以减少图像编码需要的 50%字节数。这是因为在原始图像中需要 12 个值来描述 2×2 的方格，在压缩图像中仍需要 4 个值来描述亮度或 Y 值，但分别仅需 1 个值来描述 C_b 和 C_r，共计使用 6 个值而不是 12 个值。 627



原始图像　　Y分量　　C_b分量　　C_r分量

图 14-31　一幅压缩图像和它的三个 YC_bC_r 分量。注意，压缩的失真可以通过在 C_b 和 C_r 图像中寻找锯齿状的斜线看到。最左边的压缩图像看上去"很完美"，即使在 C_b 和 C_r 子图中有明显的失真

14.8.3 有损压缩

将图像转换到 YC_bC_r 等与感知相关的颜色空间，进行了两种类型的压缩。首先，有损压缩阶段将人眼不能感知的信息去除——虽然信息丢失了，但是使用了不被人类察觉或很少能察觉到的方式。例如，我们已经介绍颜色信息降采样的好处，在有损压缩阶段之后，压缩系统进行无损压缩来去除统计上冗余的信号。

眼睛的敏感程度通常用感知不同频率的能力来描述。在大约每个视度的 6 个周期之外，我们感知模式的能力迅速下降⊖。因此，一个有效的压缩图像的方法是将图像内容按频率排序，只保留低频的变化。

图像经常按照它的频谱内容描述。图像或者图像的一部分可以利用离散傅里叶变换（Discrete Fourier Transform，DFT）分解为频谱分量。DFT 将图像表示为空间正弦曲线的加权叠加，如图 14-32 所示。根据图像（或者图像的一部分）内容，对不同频谱分量赋予不同的权重。需要特别注意的是，一个 256×256 大小的图像转换为一个 256×256 的频谱权重数组，当执行离散傅里叶逆变换时，可以在浮点精度范围内恢复原始图像。

因为我们的眼睛对低频空间频率最敏感，所以我们的目标是用较高的精度来传送这些频率的系数。这种频谱分析可以使用离散余弦变换（Discrete Cosine Transform，DCT），与上面描述的 DFT 相关，它确保最重要的频率以最高的保真度传输。作为基于 DCT 的图像压缩算法的一部分，图像首先被分割为 8×8 像素的图像块，用来完全覆盖图像。选择图像块大小为 8×8，是因为块的大小需要是 2 的幂次方，8×8 这个大小是在低复杂性与覆盖图像中足够大的有用面积间的一个合理折中。

DCT 采用 64 个不同的基函数来表示每个图像块中的像素，每一个基函数表示水平和空间频率的组合。图 14-33 给出了这 64 个基函数。我们可以计算一个图像块的 DCT，生成 64

⊖ 在 1 个周期中，图像的一部分从明变暗，再从暗到明。1 视度与从眼睛到所看到的物体角度的 1 度对应。

个系数，每个系数对应于一个基函数，再计算一遍 DCT，会得到原始图像，只有无关紧要的舍入误差。什么也不会丢失 [212]。

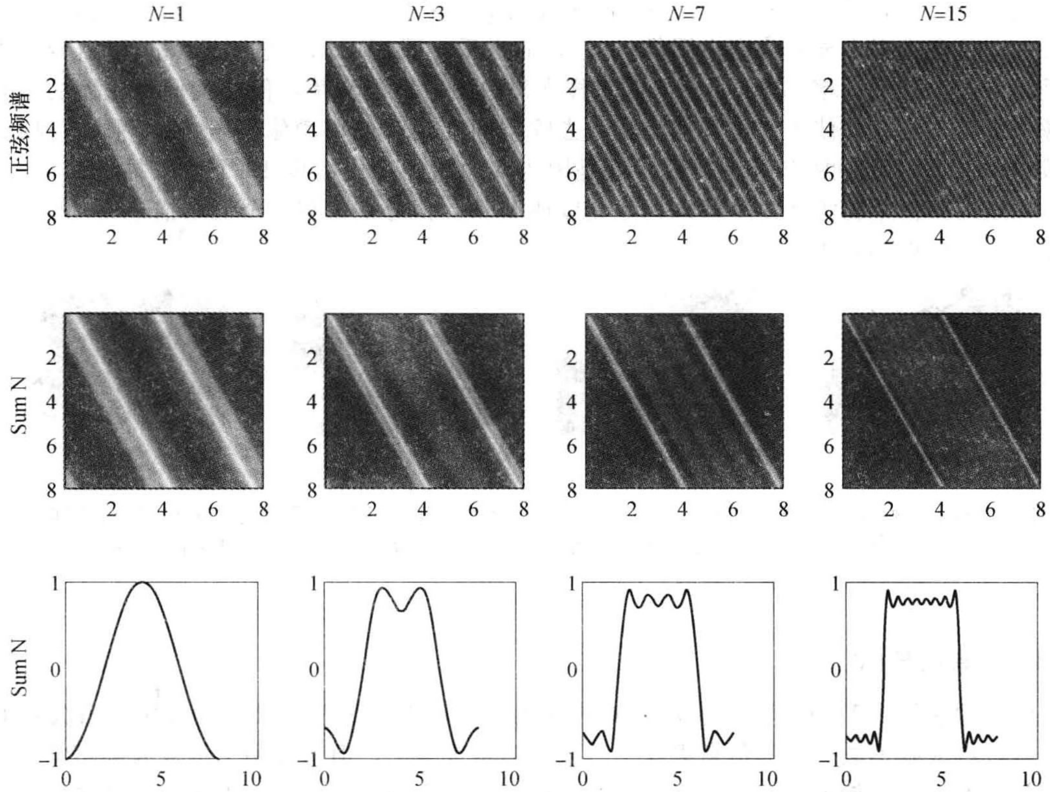


图 14-32 几个不同的空间频率以及它们如何组合表示任意数据。最上面一行给出了 4 个简单的不同空间频率的正弦曲线，每个图像宽度包含 1～15 个周期。根据合适的权重将前 N 个正弦曲线相加，我们得到如中间行所示的图像。当 N 增大时，我们可以更容易地创造白和黑之间的锐利转变。最底行清晰地显示了中间行图片的中间水平切片的亮度。这是频谱分析的基础

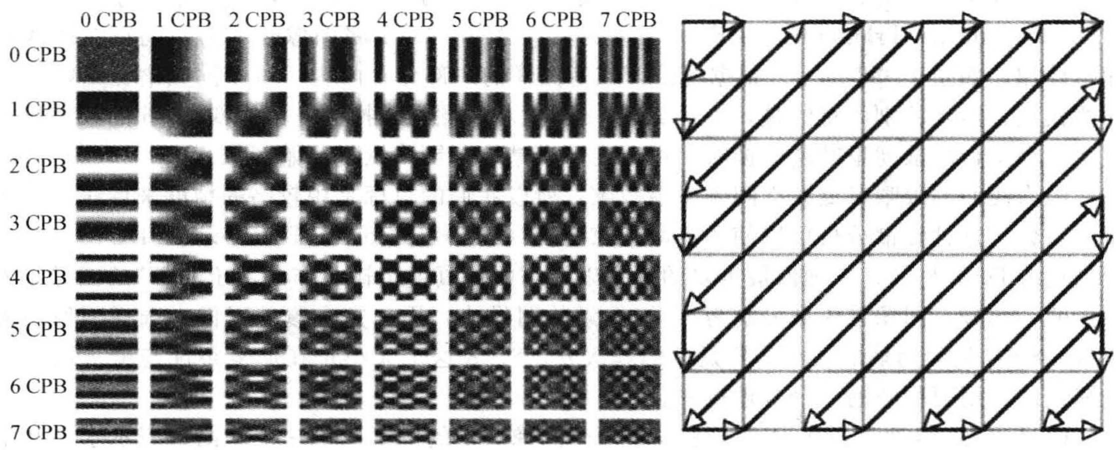


图 14-33 左边显示了用于 8×8 DCT 的 64 个基函数。给出了每个水平或垂直块（CPB）的 DCT 周期。右边给出的扫描方式表示了这些块如何排序，使得最重要的数据，即（0，0）DCT 系数最先传送

14.8.4 无损压缩

上面介绍的有损压缩技术通常后面跟着无损压缩技术——进一步压缩数据，但是不在表示上引入任何误差的技术。一旦我们去除了信号中的感知冗余，我们的目标就变成消除数字中的统计模式。这个领域常用的两种方法是游程编码（Run-Length Encoding，RLE）和熵编码（entropy coding）。 628

作为 DCT 处理的一部分，我们把在 8×8 DCT 图像块中的像素根据它们的重要性重新排列。我们将 8×8 图像块中靠近右下角的系数去除，因为它们是高频分量，很难看到，能量也低，因此不重要。因为邻近的值是相近的，即使在 DCT 之后，因为它们相关。——我们可以通过只传递差值来提高效率。然后我们使用游程编码，通过传递值和这个值出现的次数来更有效地传递这些系数。

压缩的第二个阶段称为熵编码。熵编码使用系数的熵（随机性）来设计最优的编码策略。在文本领域，Lempel-Ziv 是常用的压缩字符串字符的方法，解压缩后可以得到与原始文本无差别的内容。在图像压缩中，霍夫曼编码使用最少的位数传送每个符号，仍然可以准确地重建游程编码的原始字符串。 629

14.8.5 时间冗余

去除冗余是压缩的关键。在图像编码中，我们通过使用有损压缩技术，忽略人眼不能感知的图像细节来去除冗余。然后，我们使用两种无损编码，游程编码和熵编码，高效地传送剩余的信息。在视频压缩中，我们可以去除两种额外且相关类型的冗余：运动估计（motion estimation）和图像预测（image prediction）。

运动估计和图像预测在视频中很重要，因为视频中的一帧与接下来的一帧通常很类似。我们可以只传输两帧之间的插值，称为增量图像，以提高压缩视频的能力。在理想状态下，我们可以只传送场景中的第一副图像，然后传送增量图像，根据之前的数据重建图像。这意味着，中间图像只有在我们先解压缩完之前所有图像后，才能进行重建。此外，这种方式对错误很敏感，使得在视频中进行跳跃更困难。

MPEG 压缩提供了另外一种方式，其中增量图像可以通过与前向或者后向相关的完全传送的图像计算得到，完全传送的图像称做 I 帧，我们将在 14.8.7 节中进行介绍。 630

14.8.6 运动预测

给定两幅图像，我们对第一幅图像利用 14.8.3 节介绍的方法进行压缩，然后用压缩的图像来预测第二幅图像，仅传送它们之间不同的部分。完成这个工作最简单的办法是使用第一幅图像的像素来预测第二幅图像中相同位置的像素。因为视频帧的改变和物体的移动，仅使用差集不一定能有效地降低数据传输量。

因此，视频压缩使用了更有效的方法叫做运动补偿（motion compensation）。在运动预测中，我们基于在之前图像中附近图像块的像素来预测在新的帧中每个 16×16 图像块的像素。这就引入了在之前帧中查找最佳匹配的问题。换句话说，我们要查找用 Δx 和 Δy 表示的位移，使得差异函数 $E(\Delta x, \Delta y)$ 在连续的两帧 I_1 和 I_2 间最小，表示如下：

$$E(\Delta x,\Delta y) = \sum_{x,y}(I_1[x+\Delta x, y+\Delta y] - I_2[x,y])^2 \tag{14-13}$$

因此，图像中每个与邻近图像块具有很高（或者足够）相似度的 16×16 图像块都用预测位移函数表示，它只需要很小的位数。值得注意的是，虽然这个过程叫做“运动预测”，

但算法实际上是寻找相似的像素块，利用它来更容易地描述压缩的视频。最佳值可能也可能不能反映图像中潜在的运动。

可以独立（即单独）于或者相关于邻近帧压缩视频流中的帧。这些单独压缩的帧称做“内部”帧或 I 帧。在压缩流中，I 帧出现的频率取决于应用。更具体地说，它取决于应用可以忍受的延迟。例如，在数字广播应用中需要高的刷新率来支持编辑和浏览，相比之下视频会议应用则不需要。I 帧的比率与压缩没有必然联系。在视频流中，I 帧对搜索起到了关键作用，因为大多数流应用仅能搜索到最后一个 I 帧。与流不同，直播流中的 I 帧频率较低。这就是为什么数字频道不能立即切换的原因。

图像中每个 16×16 图像块可以通过分析和保存预测误差来压缩：

$$I_e(x,y) = I_t(x,y) - I_i(x+\Delta x, y+\Delta y) \tag{14-14}$$

其中，I_t 是视频流中当前的帧，I_i 是参考帧，即我们用来作为参考的已解压的帧，Δx 和 Δy 是对宏块的运动预测向量，I_e 是 16×16 图像块误差。重要的是，I_i 是解压缩的（decompressed）帧，因为所有接收者可以访问到，即他们不知道在有损压缩阶段产生了什么误差。这也就是为什么 I 帧作为参考帧的原因，因为编码器和译码器不需要同步就可以参考图像。在最好的情况下，图像误差 I_e 是全 0，因此仅需要传送运动预测向量。一般来说，误差不是 0 但是足够小，可以很好地压缩。当误差很大时，意味着运动预测向量不能很好地预测新帧中的值。这种情况下，系统规则会丢弃运动预测向量，传送整个帧作为 I 帧。所有这些方法
631 都是在 14.8.7 节中讨论的 MPEG 压缩标准的一部分。

式（14-13）和运动预测所蕴含的方法是清楚的，但是它掩盖了三个重要的细节。第一，求和通常是在一个宏块（图像中 16×16 的区域）上进行的，估计了在其他图像中最匹配的像素值。因此，对于每个补偿的计算需要花费 256 次乘法和加法。第二，直接实现这种计算的代价，随着我们考虑的最长距离呈平方级增加。搜索 2×2 窗口是简单的（但是不容易找到匹配），但搜索 32×32 窗口的代价很高。因此人们通常使用智能搜索策略来搜索尽可能大范围的移动，以避免无效的计算。第三，对于每一个宏块而言，最好的运动预测向量都是与其他宏块独立的。为每个宏块查找最佳预测的计算，通常与物体运动类似，但是这些计算忽略了我们在 14.2.2 节提到的窥孔问
632 题。两个图像之间运动预测向量的例子见图 14-34。再次说明，运动预测这个名字有些误导。

14.8.7 MPEG 标准

如今，许多多媒体信息检索系统中的音-视频数据使用 MPEG 编码，它是压缩和传输多媒体信息的一种标准，由国际标准化组织（International Standards Organization，ISO）和国际电工技术委员会（International Electro-Technical Commission，IEC）创建。它不只是一个标准，而是为了解决视频市场的新兴需要，随着时间演变而来的一系列标准（MPEG-1、MPEG-2、MPEG-4、MPEG-7 和 MPEG-21）。

每个标准的详细描述本身就可以构成一本书。本节仅包括了绝大多数 MPEG 标准中共同的基础原理，并突出了区分每个标准的关键特征。我们从 MPEG-1 和 MPEG-2 开始。

1. MPEG-1

MPEG-1 标准从 1988 年开始，1992 年年底获得批准。虽然，已经有一个视频压缩标准 H.261，但是它的质量低，并且不支持交互，而这是游戏行业的一个关键需求。1988 年，数字视频不适合通常的存储介质，因此像 Video CD 和 CD-ROM 这些应用也促进了 MPEG-1 的发展。最大的挑战是将音频和视频存储在专门用于音频的存储介质上。此外，需要交互性来支持随机访问。

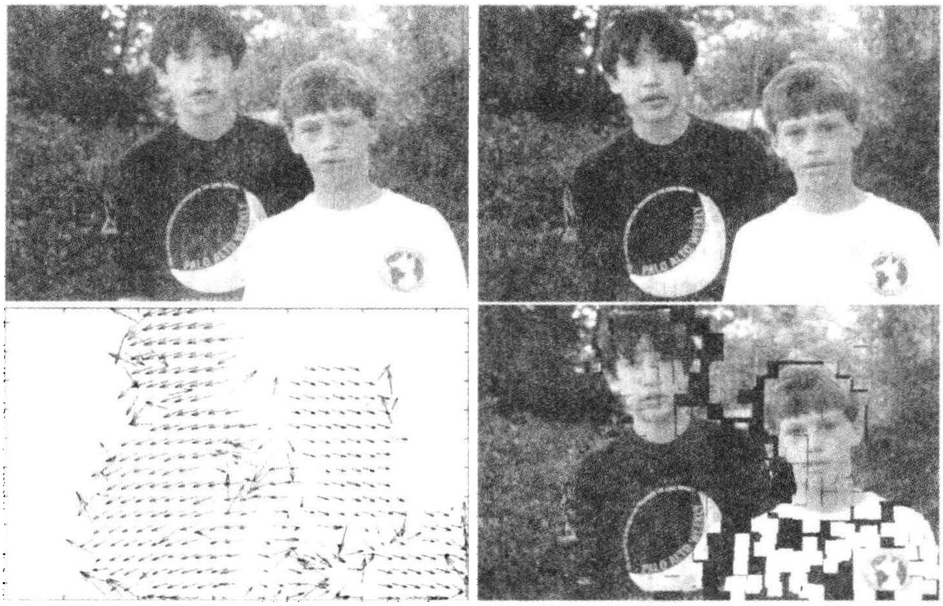


图 14-34 显示了运动预测向量。顶部显示了原始帧——一个男孩移动到左边，另一个移动到右边。左下图片给出了对于每个 16×16 图像块，计算得到的运动向量。要注意的是白色 T 恤的左下部没有可靠的方向，这是因为没有足够的纹理来可靠地估计运动。最后，右下图像给出了第一幅图像使用估计的运动向量转化得到的结果，看起来很像第二幅图像

MPEG-1 的视频质量可以与 VHS 录像技术的视频质量相当，传输率为 1.5Mbps，其帧的大小为 352×240，每秒 29.97 帧，伴随 192bps 的立体声。总之，它提供了一个高效的压缩算法，该算法使用当时（1993 年）的硬件可以实时解码。MPEG-1 已经得到广泛采用，在大多数计算机和 DVD 播放机上可以播放。3 级 MPEG-1 表示为 MP3，构成了最流行的音频压缩标准。最后，它还用于亚洲最流行的视频发布格式 VideoCD。

2. MPEG-2

MPEG-2 的开发是为了响应类似宽带和高清电视等应用，它们需要比 MPEG-1 更高的质量和带宽。MPEG-2 标准对于宽带提供的位速率范围是 3～15Mbps，对于 HDTV 提供 15～30Mbps。MPEG-2 和 MPEG-1 共享了很多视频编码单元。它基于 MPEG-1，但它是为了压缩传送数字广播电视而设计的。与 MPEG-1 相比，最显著的扩展是它可以高效压缩隔行扫描视频的能力。MPEG-2 很好地扩展到了 HDTV 所要求的分辨率和位速率，使得 MPEG-3 标准没有必要。MPEG-2 解码器也可以解码 MPEG-1 位流。与 MPEG-1 不同，它提供了多通道环绕立体声编码。此外，MPEG-2 提供了一个简单类型和一个主要类型。简单类型的目标是不能承受大延迟的应用，例如电视电话会议。在这种情况下，此类型不包含任何后向预测。这就意味着，延迟会较小，因为不需要在传输中重排帧的顺序［711］。 633

MPEG 使用周边区域的信息来压缩一帧中特定的区域。运动矢量捕获目标区域的移动，使得预测更容易。预测不仅意味着查看之前的帧。实际上，使用了三种帧的类型：I 帧、B 帧和 P 帧。I 帧表示“内帧”（intra frame），而 B 帧和 P 帧表示“外帧”（inter frame）。I 帧不需要参考其他任何帧。它们只是简单地作为静态图像编码。因此，解码可以从任意 I 帧开始。我们说 I 帧提供了视频流中的锚点，因为它们构成了随机访问的入口点，以及错误恢复的同步点。例如，在视频流中大量丢失包的情况下，通常不是试图从这种严重错误中恢复，而是跳过这段流

直接到达下一个 I 帧。从错误恢复角度来看，每一个 I 帧提供了一个新的开始。

P 帧使用前向预测来压缩和重构。它们的重构需要之前的 I 帧或者 P 帧。从前面的某一个 I 帧或者 P 帧，结合运动预测向量，我们可以计算得到新的帧。

B 帧或双向帧是独特的，因为它们不仅需要前向而且还需要后向预测。B 帧的重构需要最靠近的前一个 I 帧或者 P 帧，还有最靠近的后一个 I 帧和 P 帧。MPEG 预测可以从编码器和解码器的角度来描述。图 14-35 给出了一个在编码流中的典型帧序列，以及帧间的依赖关系。编码帧序列 IPBBPBBBI 显示在图中最上面一行。相对应的显示序列和帧顺序在下面一行中给出。箭头给出了前向和后向预测。

给定帧依赖关系，帧的编码/传输序列一定与显示/回放序列不同。否则，解码器要暂停重构 B 帧直到参考的 P 帧或者 B 帧到达。在图 14-35 中给出的显示序列可以按照 IBB-PBBBI 传送，或者等价的帧编号 1423856711910。解码器需要三个缓冲区，一个是为了前向预测，一个是为了后向预测，另外一个是为了图像重构。在 P 帧中的每个块可以是内编码的或者预测的。类似地，B 帧中的每个块可以是内编码的或预测的（前向、后向
634 或双向）。

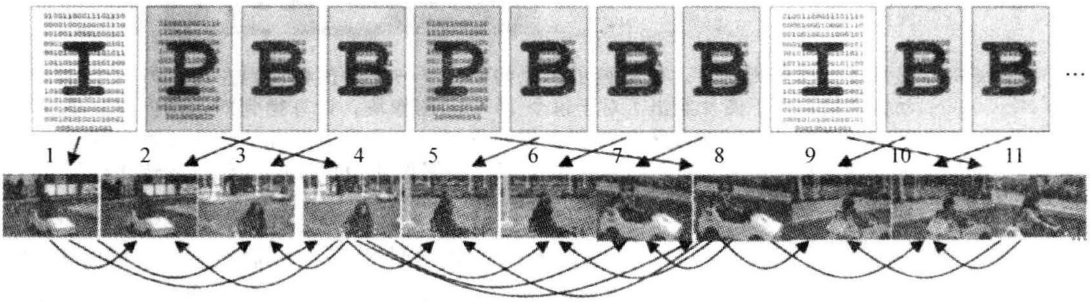


图 14-35 显示了 I 帧，B 帧和 P 帧如何用来表示一个帧的序列，以提供有效的帧间预测和随机访问。最上面一行显示了为了传输重排后的压缩帧。最下面一行给出了图像的正确顺序，弧形箭头给出了每个帧对于 P 帧或 B 帧预测的作用

3. MPEG-4

MPEG-4 是一个压缩标准，其最初目标是低位速率视频通信应用。幸运的是，它的应用范围扩展了。MPEG-4 可以在大范围的位速率下工作，范围从几千位/每秒到 10Mb/s，因而具有很好的扩展性。它可以使用基于对象的压缩，是超越块压缩的第一个标准。它从 1998 年开始，直到现在还在继续开发，是最雄心勃勃的标准之一。MPEG-4 的愿景是提供网络媒体和传统媒体之间的桥梁，并成为互联网流媒体标准。MPEG-4 支持与场景中对象交互。它支持自然和合成媒体混合。它提供了与 MPEG-1 和 MPEG-2 相同的绝大多数特征，提高了编码效率。此外，它还提供了语音、音频和视频压缩。MPEG-4 有很多等级和类型。H.264 是一种视频压缩标准，也称为 MPEG-4 Part 10，或者 MPEG-4 高级视频编码（Advanced Video Coding，AVC）[1354]。

4. MPEG-7

MPEG-7 是第一个与压缩无关的 MPEG 标准，它与媒体语义相关 [1080]。它描述了关于内容的元数据，而不是内容本身。从这方面讲，它可以看成是一个内容描述标准。它被描述成“关于数据的数据”（the bits about bits）。MPEG-7 规定了一系列描述方案、描述符、指定描述方案的语言和描述符编码过程。这种语言是描述定义语言（Description Definition Language，DDL），它使用 XML 定义。用来计算 XML MPEG-7 方案中实例值的算法并不

属于这个方案的范畴。例如，MPEG-7 有一个描述方案用来描述组成视频的片段或镜头列表，但是，用来计算得到这个列表的分割算法或者场景切换检测并不是 MPEG-7 标准所要考虑的。这与 MPEG-1 用来计算运动向量的算法类似，它取决于具体实施者的技术。

5. MPEG-21

最后，MPEG-21 是一个对于多媒体交付和消费的开放框架。它用来应对多媒体制作人员在描述多媒体知识产权时所面临的挑战。在 MPEG-21 中，交易和发布的基础单元是数字条目（Digital Item，DI），即音频、图像、视频和文本元数据的组合，它们包含了这些部件间的关系。MPEG-21 定义支持数字条目交易的技术，因此用户可以高效无缝地与它们进行交互。版权表示语言（Rights Expression Language）是一个在内容提供者到消费者多种角色间共享数字版权信息的标准。

MPEG-21 目标的最简单形式是提供一个框架，使得两个人可以在其中高效平滑地互相交互、操作、贸易、消费和访问数字条目。希望这种透明的交互可以阻止非法文件共享。 635

14.9 趋势和研究问题

多媒体信息检索在过去三十年中，经历了至少三个不同的阶段：基于内容的、基于文本的，以及混合的方法。

学术界和工业界几十年来进行了大量的努力，目标都是了解图像的内容，使计算机可以生成索引，以便进行搜索。由于许多上文所述的原因，产生了语义鸿沟，使得这些努力都没有成功。许多失败的尝试包括 IBM（利用 QBIC 系统进行图像搜索）、Virage 和 Musclefish（对于声音）。

2001 年，谷歌引入了图像搜索引擎，使用网页上图像周边的文本来提高搜索结果。这是一个聪明的填充语义鸿沟的方法，因为网页上的文字通常描述了伴随的图像。为了使这个方法取得好的效果，文件名和与网页链接紧密相关的锚文本提供了最好的信息，应给予较高的权重 [403]。这些信息使大规模 Web 搜索引擎可以使用人们在 Web 上产生的文字自动地标注图像。

这种基于文本的 Web 图像搜索方法又在两个方向上进行了扩展。第一个方向，Flickr 等网站鼓励人们上传照片并使用几个单词标记它们，描述它们的内容。搜索就变成了查找与查询相符的图像标签，并且对这些标签进行排序。照片和它们的标签可以进行聚类（见图 14-36），但是图像的相关度仍然是一个问题。Flickr 对图像根据时间和称为兴趣度的分数进行排序。 636

第二个方向，ESP Game 等网站鼓励人们通过竞争标注图像 [1644]。向两个随机选择的用户显示一幅图像，用户独自给出建议的标记词。竞争体现在两个用户试图挑选对手心中所想的单词。每个用户不会看到对方猜测的单词，直到出现相符合的单词为止，也就是说，一个用户输入了对手已经输入的单词。因为图像是两个用户唯一共享的内容，所以两个用户所共同挑选的词对于图像是好的标签。通过将同一幅图像给不同的用户对，一定程度上可以确保标签是真实的，反映一种共识，而不只是一个侥幸的游戏。

基于内容的搜索方法完全基于自动提取的图像特征，而基于文本的搜索方法完全基于人们建议的单词，它们代表多媒体检索中的两个极端。显然，中间的某种方法是更适合的。正如上面所讨论的那样，人脸识别、语音识别、光学字符识别，以及其他一些有监督的自动内容识别和提取方法，表明计算机可以有助于减少多媒体检索中的语义鸿沟。



图 14-36 Flickr™ 中的图像根据标签聚类，给出了老虎、计算机、花和蝴蝶的图片

14.10 文献讨论

本章只提供一些多媒体检索中重要话题的介绍。本节介绍一些在本章中没有介绍的重要方向。这只是一个个人列表，包含了我们认为最重要的和有发展潜力的方向。我们希望这将作为读者进一步阅读的起点。首先，介绍几本关于多媒体检索的书 [932，1101，1392]。

多媒体检索的讨论都会提到用来处理大量数据的机器学习工具（参见 8.2.1 节）。Richard Duda 关于模式识别的书 [517] 和 Chris Bishop 关于机器学习的书 [205] 都是找到基础知识的好地方。书中的很多工作使用了产生式模型，它可以用于根据概率生成每个类的所有数据。但更好的用于模式分类的技术是判别式方法，例如支持向量机（SVM）[1439]。最近的工作，大量采用了弱分类器，每一个具有不同的类型，解决不同类型的错误，用它们来构造分类器树或森林，其结果往往胜过一个复杂的大分类器 [339]。

另外两种技术的重要性在多媒体检索中也在上升。首先，语音识别开始处理多媒体数据，且未来将更加重要。Huang 的书对 HMM 技术做了很好的介绍 [788]。其次，大规模的多媒体数据库意味着，需要局部敏感散列（Locality-Sensitive Hashing，LSH）等随机算法来快速找到需要的内容 [1490]。这些理解数据、分类数据和识别内容的自动方式对于已经保存了数十亿幅图像和数百万小时视频的数据库是重要的。

图像处理中的关键步骤之一是从纹理中提取信息。有许多基于视觉感知模型、统计特征，以及 Gabor 小波特征 [779] 等过滤器的纹理检测方法。复杂的纹理测度包括基于感知的测度，例如对比度和粗糙度 [1555]，基于 Gabor 滤波器的频谱测度 [615] 和基于 Wold
637 分解统计的测度 [1038]。

用户有这么多的内容可用，推荐系统就变得很关键。数百万视频可以提供给用户，系统下一次应该显示哪一个？这是一个关键问题，因为用户有很多内容的来源，一次不好的推荐

就会造成用户去其他地方寻找。一个流行的方式就是，根据类似用户的访问模式来进行推荐，这种方法叫做协同过滤（collaborative filtering）[15]。有些购物、音乐和视频网站已经采用这种方法。最近，Netflix 挑战已经激发了许多研究人员。由 Bell 所完成的获胜工作，描述了基于很多不同信息源的最新方法 [172]。协同过滤的下一代工作需要处理非常稀疏的数据 [134]，并考虑内容分析（如 Agarwal 所描述的 [16]）。最后，相比仅根据内容的答案而言，大量的用户评价往往可以提供比基于内容的方法更好的答案。这是基于项间相似度系统的目标 [1494]。

理解声音信号的内容往往是理解多媒体内容最简单的方法。音频是随着时间推移的一维信号，对于语音而言，语义鸿沟较小。正如上文中所讨论的，语音识别是第一步 [788]。但是，如 14.4.2 节中所介绍的，语音信号通常受到背景音乐、干扰的说话人或者环境噪声的破坏。人们很容易理解这些声音，并把它们忽略掉。利用计算机来分割独立声音的过程称为计算听觉场景分析（Computational-Auditory-Scene Analysis，CASA）。Wang 的书 [1665] 给出了很好的关于目前技术的概述，现在需要更多的工作将 CASA 与多媒体分析进行关联。

在音乐方面，音乐检索有一个活跃的社区。每年的 ISMIR 会议 [1337]，讨论从音乐图书馆到音乐内容分析在内的所有问题。此外，《IEEE Transactions on Audio，Speech，and Language Processing》的专刊 [1491] 和 Casey[340] 以及 Orio 的文章 [1232] 都很好地概述了目前的工作。

许多研究人员利用不同的方法分析视频摘要和浏览。Xiong[1729] 定义了摘要的两个主要类别：基于目录的（Table-of-Content，ToC）和基于加亮的。这些类别分别对应于自顶向下和自底向上的摘要生成方法。PanoramaExcerpts[1561] 结合图像拼接和关键帧，创建了视频故事板。对于每个镜头（片段）显示的结果图像有不同的尺寸。一个特殊的背包算法用来优化版式同时保持视频内容的时间线。场景转移图（Scene Transition Graph）[1747，1748] 试图用图来显示视频基本故事结构，以捕获视频语义。

不论是图像研究人员还是视频研究人员都在努力标注内容。图像分类领域里两个常用的数据库是 Caltech 101 [551] 和 Caltech 256[677]。在这两种情况下，目标都是要在 101 或 256 类中找出图像内容。这些图像经过很好的构造，只包含一个单独的可识别对象，因此这个任务比在大规模 Web 数据库中查找要容易。然而，问题仍然是非常棘手的，在写本书的时候，最成功的方法是由位于印度的 Microsoft 公司提出的 [1629]。 638

视频识别和视频分类工作使用 TRECvid[1495] 数据库。这些数据库很重要，因为它们为比较研究算法提供了一个公共基准。在竞赛中，两个更重要的任务是检测视频中的概念，例如主播或户外场景，以及通过人工引导过程来找到回答特定查询的内容，例如一个人在双翼飞机机翼上行走。关于基于内容的视频检索的近期综述可以参见 [1498]。

在高维空间中找到最近邻，例如音乐的图像描述，是很困难的问题。一种解决方案是局部敏感散列（LSH）。这是一种常用的查找重复网页的算法。通常，在散列算法中，相近的字符串分布得很广，因此散列桶碰撞就很少发生。在 LSH 中，许多函数将高维空间的向量映射到一维的线性桶中。与查询向量出现在相同桶中的向量具有很高的匹配度 [1490]。另外一种方法是学习一个半监督映射，将相似的声音放到相似的散列桶中 [133]。

然而，上面介绍的所有使用内容分析的工作还都没有成功的产品。取而代之的是，现在大多数流行的系统使用社交网络来帮助人们找到他们想要的内容 [901]，使用用户生成的标签来简明地描述内容 [516]。标签可以描述对象的地理或时间信息 [900]。它们的统计信息可以用来自动推荐新的标签 [1477] 或解决二义性问题 [1677]。

最后，搜索中最困难的问题之一是确定相关性和排序。Flickr 中包含了 10 万幅标记了金门大桥（GoldenGate）的图像，将哪 10 个最先返回给用户？类似的问题也出现在文本中，不过文本中有 PageRank[263] 等基于全局链接结构来确定不同网页权重的算法。但是多媒体领域还没有出现类似的算法。

有两个有希望的方法——两者本质上都是在使用关键词检索之后，查找出现在图像概率分布峰值上的图像，以缩小检索范围。对于旅游图像的方法是由 Kennedy 提出的 [902]。Jing[835] 构建了邻接图来查找相似图像。然后通过使用类似 PageRank 的算法找到图中心的图像。近期工作考虑使用基于重排 [782] 或机器学习排序 [211] 的方法学习正确的多媒体检索结果排序。

然而，我们需要更多的工作来帮助用户与互联网上所有的新媒体建立联系。现在的媒体数据库已经包含了数十亿条条目。标签是不完备的，许多人也并没有链接到社交网络。标签在 Web 上将只有有限的应用。一个用户关于圣诞鹅的图像与其他用户每年圣诞节与 Bob 叔叔走在沙滩上的图像看上去是十分不同的。Web 上的隐藏媒体，其中一些是很有价值的，在等待下一代多媒体检索研究人员去发现。 639

企业搜索

——David Hawking 著

15.1 介绍

和图书馆一样，企业、政府机构以及非盈利性组织必须处理不同媒体和格式的文件，而许多信息对于机构来说是独特的和专属的。机构的一些信息资源保存在关系数据库或专门的应用程序中，但很多数据都是非结构化文本类型的，需要设计信息检索系统来处理它们。

在机构内部查找信息的信息检索技术称为*企业搜索*（enterprise search）。企业搜索可以解释为对一个机构所拥有的数字文本材料进行检索［719］，包括搜索它们的外部网站、公司内网以及它们持有的其他电子文本，如电子邮件、数据库记录和共享文档等。

很多时候，企业搜索工具的用户觉得他们的体验无法与 Web 上的体验相比。“我可以很容易地在 500 亿个 Web 网页中找到我曾祖父的出生证明。为什么无法在我的小公司中找到去年的财务报告?”有时，批评是相当刺耳的，尽管所使用的技术可能是从 Web 搜索引擎衍生而来，并由同一家公司销售和支持的，但还是会得到批评。本章的目的之一是概述企业搜索的问题是如何不同于 Web 搜索的。

本章介绍企业搜索系统的架构和系统中不同组件的功能，还讨论通过研究真实的企业搜索活动并加以刻画，从而用于科学研究的尝试。其他主题包括企业搜索的评价方法、文本检索会议（TREC）中的企业搜索研究、企业搜索系统的调试、发布与搜索间的互动以及对企业搜索部署所能期待的性能级别。此外，还涉及两个虽然不仅限于企业搜索，但对于企业搜索特别重要的主题：联合搜索和搜索情境化/个性化。 641

15.1.1 企业搜索的特点和应用

企业搜索的许多特点为信息检索设计人员带来了挑战［274，1162］。企业中的信息可能是结构化的，也可能是非结构化的。文档有很多来源，也许是不同语言的，通常没有格式标准。元数据可能通过某些不同的模式产生，或者可能根本没有元数据。并非所有的用户对于所有的信息都有相同的访问权限，员工记录等一些信息是高度机密的。联合不同信息库的需求意味着对于不同来源和不同格式的数据必须建立一个单一的排序列表，而不同的情境可能需要不同的排序方法。也就是说，除了简单的索引和查询过程，企业搜索工具还必须执行很多功能。基本的搜索工具不用做这些工作。

基于 Web 的经验，我们期望对企业信息的查找应该是快速和高效的，应该通过单一的界面完成。然而，在机构内对这些期望通常无法满足，有证据表明员工要花费大量的时间用于搜索，但往往无法找到所需要的信息来执行他们的工作。例如：

- 据国际数据公司（International Data Corporation，IDC）报告，一家拥有 1000 名信息工作人员的公司，由于较差的搜索，预期每年浪费超过 500 万美元的工资。他们报告说，人们每周花费 9～10 小时来搜索信息，并且有 1/3～1/2 的情况没有成功［805］。

- 据Butler集团报告，公司工资的10%在无效的搜索中浪费了[526]。
- 埃森哲（Accenture）在2007年对于1000个中层管理人员的调查中发现，他们每天花费2小时进行信息搜索，而他们通过搜索得到的信息超过一半是无用的[713]。

另一个在财务上受企业搜索影响较大的领域是电子信息发现领域。为了支持与高价值诉讼关联的发现活动[1378]，机构越来越需要能够对其内部所有信息源以可审计方式搜索的工具，即使这些搜索是由外部的专业人员完成。此外，高效地搜索机构的对外网站对于机构的使命来说至关重要，无论是传播信息、支持政府活动、匹配职位申请者和空缺职位，还是网上销售。电子商务网站通常由搜索工具驱动，其功能包括支持搜索产品信息和评论、实际产品的真实购买页面、搜索驱动的广告和智能推荐。因此，外部网站上的企业搜索软件对于客户和利益相关者的利益来说是个高价值的信息源。此外，他们产生的查询数据可以提供与客户或社区兴趣相关的趋势和突发情况的信息，并识别出尚未满足的需求。事实上，有些领
642 先的企业搜索工具可以提供详尽的报告能力。

企业搜索工具也执行其他的功能。搜索引擎通常给机构网站提供导航链接、RSS订阅、分面浏览和分类显示等功能。当多个项目团队组合到一起时，搜索工具在机构内部越来越多地用于专家定位。此外，自动生成的内部报告可能包括从搜索结果中得到的摘要。最后，Web发布和搜索在几乎所有的公司内网中都会得到充分利用。

企业搜索几乎一定要包括一个小规模的Web搜索维度。Upstill等人[1619]表明锚文本和PageRank的变体在小规模Web环境中是有效的，虽然简单地对入度计数就可以得到PageRank的大多数优点。Hawking等人[720]研究了利用机构外部链接增强机构网站搜索质量的可能性。他们发现所研究的大多数机构的外部链接往往引用网站的入口页，只有一小部分链接其他的目标。因此，它们对于解决特定网站的内部问题的价值是微不足道的。Hawking和Zobel[724]研究了主题元数据在回答由用户提交到机构网站的查询时的价值，并且关注于元数据标记。由于内在的局限性和实现不力的情况（尽管资源保证），他们发现主题元数据在回答查询中只有极小的价值。虽然已经有一些技术从Web搜索转化成企业搜索，但这两个应用还是在很多重要的方面存在差异，我们将在本章介绍。一个主要的不同是，在机构内制造垃圾没有经济方面的回报。

15.1.2 企业搜索软件

企业搜索软件已经使用了一段时间，某些早期的系统是信息检索科学研究的衍生产品。有些公司的企业搜索产品是很著名的，如FAST Search&Transfer[1]（2008年被Microsoft收购）和Autonomy[2]（2005年收购了Verity，2009年收购了Interwoven内容管理系统(Content Management System，CMS)技术）。IBM、Oracle和谷歌等各大搜索和软件公司也针对这个市场开发了产品。谷歌的Search Appliance由于其易用性和用户对于谷歌的熟悉而变得流行[70]。

提供企业搜索产品的规模较小的公司可能通过提供特殊的功能得到合适的地位。Vivisimo[3]是提供聚类输出的搜索引擎开发者，他们也有针对公司市场的企业搜索产品。Endeca[4]

[1] http://www.fastsearch.com/。
[2] http://www.autonomy.com/。
[3] http://vivisimo.com。
[4] http://endeca.com/。

提供了带有“引导导航”功能的产品，提供了可能的搜索过滤器作为结果界面的一部分。Funnelback[⊖]专门从事用于企业、网站和门户网站的“软件即服务”（Software as a Service，SaaS）。 643

15.1.3 工作场所搜索

将企业搜索与员工在工作时进行的其他搜索区别开来是有用的。大多数员工都可以使用“桌面搜索”功能，这或者内置于他们个人计算机的操作系统中，或者由 Copernic[⊜]或谷歌[⊜]提供。Dumais 等人［518］报告了对这种搜索类型的扩展，它包括搜索用户之前看过的所有文档（如下载的网页，收到的电子邮件等）。

一般情况下，我们可以将所有由员工执行的搜索归于“工作场所搜索”这一标签下。这个标签不仅覆盖了对企业信息、桌面保存的信息和之前浏览的信息的搜索，而且还包括对机构外部信息源的搜索，如 Web、专利数据库、法律资源、订阅信息服务等。

由于每一名员工所检索的资源集是不同的，并且机构对所有相关信息建立组合索引也是不可行的，所以对于工作场所搜索来说，唯一可行的单一搜索框方法是“个人元数据”［1580，1583］。在［1580］中的一个早期综述说明了不同员工所访问的资源是多样化的。

15.2 企业搜索任务

在非结构化信息和即兴搜索需求对机构业务的重要程度方面，不同机构有很大不同。混凝土厂商或美发沙龙对于企业搜索的需求可能很少。而内部和外部信息的搜索（不管是非结构化的还是半结构化的）对于政策顾问公司的生产力和竞争力却是很重要的。很明显，它对于国家情报机构的运作是至关重要的。在技术支持中心，搜索的有效性（对于文档和客户历史）会决定生产力和盈利能力。

15.2.1 搜索支持任务的例子

员工执行的很多任务可通过使用搜索工具而实现或者变得更高效。搜索有时由企业范围的搜索工具所支持，但在其他情况下，搜索嵌入到特定的应用中。现在我们给出一些例子，这些都是由特定应用或者集成信息检索工具所支持的任务。这个列表还远远没有完成，但对可能遇到的应用范围给出了一个建议。 644

1. 批准员工旅行请求

为了决定是否批准旅行请求，经理需要各种信息：员工是什么级别的？该活动对于员工和公司是否有益？员工去年在旅行上花费了多少？公司对于这类旅行的政策是什么？员工表现是否优秀？他们缺席工作是否会造成生产的损失或者不能满足最后期限？在这种情况下，一个新任命的经理需要搜索各种信息源，如电子邮件、人力资源数据库和企业内网的政策部分，以做出正确的决定。

2. 在呼叫中心回复电话

很多呼叫中心依靠在精心准备的文档上操作高效的搜索工具来最小化运营成本。如果搜索工具总是可以找到正确的答案页面，那么一个不太熟练业务，或者没怎么经过培训的员工就可以在低工资下工作。如果搜索工具减少了寻找答案时浪费的时间，那么客服电话就可以

⊖ http://funnelback.com/。

⊜ http://www.copernic.com/en/products/desktop-search/。

⊜ http://desktop.google.com/。

更简短，相同数目的接线员可以处理更多的电话。

3. 在争论过程中回应

当项目失败或者犯错误时，为了劝告或惩罚员工，或者决定采取什么样的姿态与外部机构进行谈判，机构可能需要进行能逆转局势的沟通。对于关键电子邮件和项目文档的搜索可能是做出正确回应的关键。

4. 写一个提案

对于一家私营公司，回应一个大的“征求方案”或者“要求招标”的机会可能是一个耗时和昂贵的业务。许多这样的标书需要对成百上千个问题进行回应，导致提案文档超过100页。如果搜索工具可以快速、准确地定位到之前的标书中能给出最好回应的段落和图片，并且定位到其他有用的当前公司文档，那么花费会大大地减少。

5. 获得并捍卫专利

Dupont、BASF 和 Pfizer 等工业公司的营收基本上是依靠他们的专利清单。在投资数十亿美元到工厂中制造新的化学品或药品之前，他们必须确定他们的知识产权是安全的。工业公司一般订阅商业专利数据库和文献服务，并利用专业的专利检索工具。专利搜索带来了许多挑战，包括：专利律师晦涩的语言；需要搜索所有语言的专利；需要搜索图表、化学结构
645 和文本；需要识别化学和生物名字的变体；需要为一些因素强加关系约束，如反应温度。关于搜索及其应用领域的信息必然是高度机密的，因为这对于投资者和竞争对手来说将是很有价值的。知识产权搜索有几种形式：

- 专利前景展望：确定某一特定领域的专利缺口，以便将公司研究投入到富有成果的领域。
- 操作自由：由该公司创造的技术是否违反其他人持有的专利？
- 新奇度搜索：新的发现是否有可能成为专利？
- 专利无效搜索：我们能否在某个领域找到先前的工作，使得我们能够击倒由竞争者持有的、阻碍我们业务的专利？

6. 向现有消费者销售

如果做到下面这些，那么向客户成功推销的可能性会大大地增加：

- 推销的目标是有针对性地解决客户的实际问题。
- 供应商表现出他们有能力、专业，且留心客户的需求。
- 供应商能确定在客户机构中谁是最有用的联系人，他们所扮演的角色是什么。

成功的客户关系管理（Customer Relationship Management，CRM）依赖于能否有效地搜索、分析以及展示与该客户有关的所有数据，包括合同、发票、销售查询、电子邮件以及支持请求。向准客户销售也可以从有效的搜索中获益，但在这种情况下检索到的信息会在企业外部。

7. 专家发现

专家发现是大型机构中的一个特定问题。在即兴问题求解或者试图整合一个项目团队时，这个需求可能就会出现。在某些情况下，一个专用的应用软件维护一张专家登记表，它以正常数据库的形式更新和查询。在其他情况下（见下面的 CSIRO 例子），可以从创建和发布的用于其他目的的信息中挖掘专家。在后一种类型的系统中，确定候选专家集是一个重要的问题。在网页集中识别符合员工模式的电子邮件地址是很容易的，但是抽取到的地址中有多少属于已经离开机构的员工，其他又有多少是行政或客服人员而不是技术专家的？

CSIRO[440] 开发了一个早期的专家发现系统原型，它将当前的员工数据库与爬取的网

页求交集。包括相关员工的名字或该员工的电子邮件地址的文字段落被抽取出来，然后添加到一个以该员工命名的代理文档中。当在专家集中执行专家查询后，对表示员工的文档进行 646 排序，然后返回排名最高的员工在员工数据库记录中的详细联系信息。在 TREC 的企业搜索任务中的后续研究展示了改进的方法，包括 Balog 等人的语言模型 [131，132]。在这种情况下，Serdyukov 等人 [1450] 展示了在机构内部识别专家时来自于外网的访问信息会有所帮助。

在专家发现系统的可用文档中，相关文本数量可能并不反映一个人的专业程度，所以，如何展示自动专家发现系统的结果对于系统是否被接受可能是个关键。公司媒体联络代表的名字和详细联系方式可能会出现在公司的所有技术文件上，而获得诺贝尔奖的科学家可能在 Web 或者内部电子文档中很少出现。

8. 运营电子商务网站

零售商、餐饮供应商、旅行社和就业服务等企业的部分或全部的收入依赖于电子商务网站。典型的电子商务网站提供了产品搜索功能，以及查询推荐、分面导航和自动生成的交叉销售建议。电子商务网站的排序算法必须考虑到各种非传统的因素，如库存水平、使用时间和不同产品的利润率，以及商品是否“减价出售”或者参与某些促销活动。电子商务网站有时是定制的数据库应用，但是它们也可能基于具有相关功能的企业搜索工具建立。Endeca、Autonomy 和 FAST 在这个领域是众所周知的。

15.2.2 搜索类型

Broder[268] 确定了三种不同类型的网页搜索：导航型、事务型和信息型（见 7.2.1 节）。这三种类型的查询都可能会提交到企业搜索引擎，例如：

- **导航型**：“图书馆”、“人力资源”、“塑料部门”。
- **事务型**：“购买停车证”、“更新借书证”、“索取花费明细”。
- **信息型**：“知识产权政策”、“在西班牙的客户”、“产品 xyz -错误 57”。

这些类别的很多子类体现在基于搜索的任务上，如 15.2.1 节介绍的。

15.2.3 研究企业搜索

研究一个你不是其员工的机构的内部搜索行为是非常困难的。在一般情况下，机构不希望自己的竞争对手知道其员工会搜索什么，甚至他们搜索什么文档集。出于这个原因，查询日志是不可能对公众公开的，甚至对于外部研究人员的研究也不行。在任何情况下，从提交 647 的查询来推测任务是很困难的。

出于类似的原因，在一般情况下，不可能允许实验人员带着笔记板跟着员工，并记录他们的搜索行为。不仅如此，观察人员的存在可能会改变人们的行为。最后，当搜索只是员工活动的一小部分时，实验人员收集有用数据的时间就会非常长，以至于无法负担得起。尽管有这些问题，但 Hertzum 和 Pejtersen[756]（工程师）、Hansen 和 Järvelin[697]（在瑞典专利局的搜索人员），还有 Freund 及其同事 [590，589]（软件工程师）已经对机构内部的搜索行为进行了研究。

对于 2007 年和 2008 年 TREC 会议的企业搜索任务 [126]，澳大利亚政府研究机构 CSIRO[⊖] 的科学传播者（Science Communicators）对两个真实任务给出了信息需求声明和

⊖ http://csiro.au/。

“相关性”判断数据，这两个真实任务来自于 CSIRO 将科学研究介绍给公众和潜在合作者的过程中。以下转述其研究任务：

1）“我正为 CSIRO 在某个重要领域中的研究撰写一个综述性的网页，例如，旱地盐碱化。请给我在 CSIRO 内部找出一系列重要的网页，它们将成为综述中链接的较好候选。例如，描述这个领域的重要 CSIRO 项目、报告、软件工具、地图和数据表的网页，这些可能对合作伙伴或公众很有用。”

2）“对于相同的综述网页，查阅 CSIRO 网页内容，并确定 CSIRO 在该主题的专家。你可以利用这样的事实：CSIRO 的电子邮件地址有这样的形式 firstname. lastname @ csiro. au。”

15.3 企业搜索系统的结构

机构内部发布的非结构化文档的数量有几个数量级的差异。很明显，有些机构几乎没有共享的电子文档，而根据 2003 年的报告 [541]，IBM 的企业网包含大约五千万个文档。

随着内部出版材料的规模和复杂性的增加，高效、恰当的索引构建工作流的重要性也在增加。图 15-1 描绘了为异质企业数据建立统一化索引的三大阶段：收集、提取和索引。

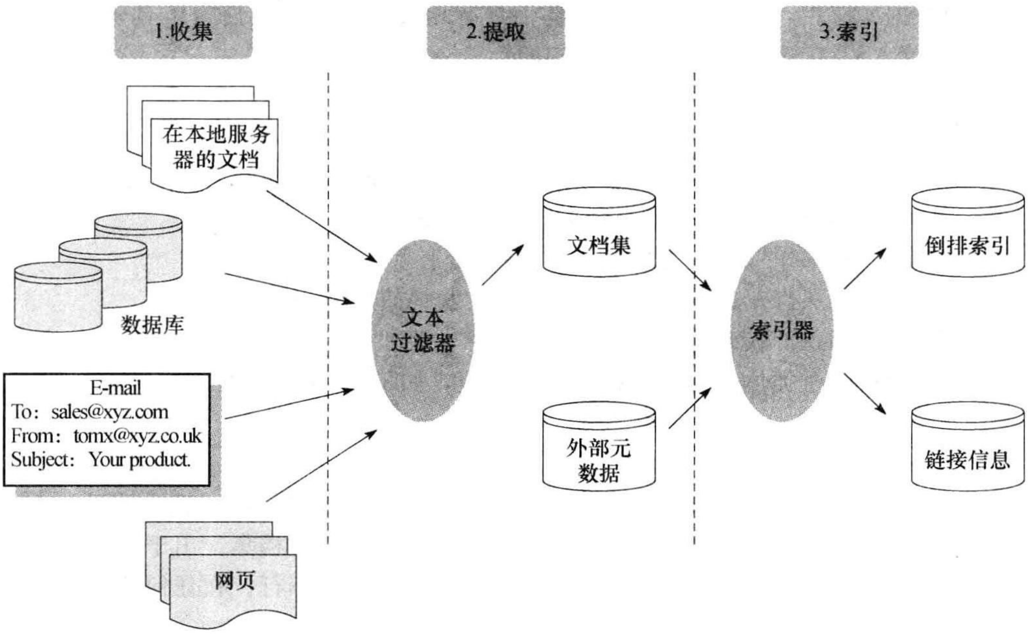


图 15-1 收集和索引工作流

15.3.1 收集

收集阶段，对应于 Web 搜索引擎的爬取（如第 12 章介绍的），可能非常复杂。第一，保持爬取的内部 Web 数据的覆盖度和新鲜度可能会面临与外部 Web 相同的很多挑战——重定向、在不同 URL 发布的相同内容的副本、识别最近改变内容的困难性、近似重复检测和
648 网络带宽问题（例如，在不同国家或城市的办公室之间），以及从 JavaScript 和 Flash 提取链接的难度。然而，值得注意的是，在企业内部可以无风险地部署某些技术，通过允许服务器提供更改内容的列表（甚至部分索引），来降低爬取的成本和持续期。此外，除权限问题

和高效找出最近更改内容的需求之外，**扫描文件系统**相对来说比较简单。此外，仔细起草的 SQL 查询可能会允许仅提取数据库收集所需的全部有用信息。

第二，在可能部署了众多**企业软件应用**的机构中，成功的收集依靠可用的、适当 API 或者适配器软件。所谓的“企业软件”，包括记录管理（RMS、EDRMS、ECM）系统、客户关系管理系统（CRM）和内容管理系统（Content Management System，CMS）等通用类别，每个类别中包含很多相互竞争的专有系统。系统中会出现一些特殊的问题。如在 Lotus Notes 中，不同类别的对象（如表格、视图、文档、导航和代理）可以通过原生 API 和标准的 Web 发布（Domino）访问。一方面，一个“文档”可以由多个内容片段合成；而另一方面，同一基本内容的多个视图可能发布在多个 URL 上。例如，所有来自某个匿名机构的 URL 实际上代表相同的文档。

- .../d/xyz \ %40. nsf/mf/3240. 1? OpenDocument
- .../D/xyz @ . nsf/b06660592430724fca2568b5007b8619/1c87d9876bc11ee8ca256fd5007722a8!OpenDocument 649
- .../D/xyz@. nsf/5087e58f30c6bb25ca2568b60010b303/1c87d9876bc11ee8ca256fd5007722a8!OpenDocument
- .../d/xyz@. nsf/w2. 2. 2/1c87d9876bc11ee8ca256fd5007722a8!OpenDocument
- .../d/xyz@. nsf/w2. 2. 1/1c87d9876bc11ee8ca256fd5007722a8!OpenDocument

从数据库中收集内容对象可以避免对重复内容建立索引，但这通常难以生成适合在搜索结果中展示的 URL。此外，标签、锚文本和用户行为数据（这可以极大地改进文档排序）等注释可能与 URL 相关，而不是与内容片段相关。

第三，在许多应用中都需要获得所收集文档的访问控制列表（Access Control List，ACL）和外部元数据（即关于文档的信息，它记录在一个单独的数据库或注册表中）。

第四，电子邮件是否可以被发送者和接收者之外的其他人检索到，这取决于企业决策。企业可能认为发送到机构地址的邮件不是员工私有的，也可能认为需要将机构和员工的电子邮件分开。在某些机构中，每一个传入的消息会归档，从而是可搜索的；在另外一些机构中，电子邮件搜索将限制在由 Exchange 等邮局软件维护的数据库；其他搜索只可能基于个人邮箱，这导致员工离职时的交接问题。

第五，有些机构热情地拥护和采用所谓的“Web 2.0”方法。如分众分类（folksonomy）标签、FaceBook 形式[㊀]的社交网络、博客、即时消息和“twittering”[㊁]，并把它们开放给员工甚至客户。一家制药厂已经热情地采用了后两种技术，并在移动设备上将它们与 SMS 短信集成。据他们说，搜索“信息流”是一个比搜索资源库更迫切的需求。

第六，收集过程可能需要很长一段时间，也为机构带来额外的大量电信资费。例如：

> **场景 1**：一个澳大利亚政府机构在 9 个省会城市都有办事处。每个办事处的员工在本地共享文件系统上创建文档。这个机构希望能够提供在所有 9 个文件系统上的统一搜索，但是，办事处之间连接的带宽在速度上和拨号的调制解调器相似。在他们的外包合同下，增加带宽将产生显著的额外费用。查询提交率则是低的。

如在场景 1 中的情况下（摘自一个真实的案例），考虑到查询量，频繁收集的代价可能并不是合理的，而联合（元搜索）方法可能是首选。

㊀ http://www.facebook.com/。

㊁ http://twitter.com/。

在所有的收集模式中，通过增量式方法通常可以大大地提高效率。在一个大型的企业网或数据库中，每天变化的内容可能连1%都不到。对一个5000万网页的企业网，每天收集其所有内容可能是不可行的或者不划算的，但是，如果不是完整扫描而只是收集变化的内
650 容，则是非常合理的。值得注意的是，增量收集的收益可能会流向过滤和索引。

15.3.2 提取

从PDF和Office文档（见第6章）等二进制文档中提取（或过滤）文本和元数据似乎应该是一个简单的工程任务。在实践中，过滤问题可能是用户对于企业搜索结果不满意的一个重要原因。这是因为过滤失败可能会导致无意义的标题、低质量的"缓存副本"和歪曲的摘要。此外，关于某个主题的关键文档甚至可能不会被识别为与查询匹配的。

为什么过滤比看起来要困难呢？第一个因素与专属的文件格式有关，如6.5节讨论的。有些这样的格式未公开发布，并且是模糊的，使逆向工程变得困难。有时，随着文档创建软件每个新版本的出现，有些细节会改变，从而增加了第三方过滤开发人员必须支持的不同格式的数目。OpenOffice和微软Office的最新版本采用了压缩XML格式，这对于过滤的易用性和准确率可能都向前迈了一大步。

第二个因素是，当用PostScript或者便携文档格式（Portable Document Format，PDF）等面向显示的格式对文档编码时，文档语义会损失。大多数文档在创建时是按照阅读顺序表示的，有一些明确标记的特殊段落，如标题、主题和表格。当转换成PostScript格式时，基本的操作是面向图形的：例如"在当前坐标系中从点（$x1$，$y1$）到点（$x2$，$y2$）画一条宽度为0.1的灰线"，"从当前字体表中，在点（x，y）打印第n个字符"。从图形到文本空间的逆向转换，一般是很困难、费时和容易出错的。

第三个因素是元数据的表示。很多知名的格式，如MSWord、PDF、OpenDocument和JPEG能够存储元数据，如标题、作者、主题、日期等。然而，在实践中，这些元数据通常是缺失的。如果存在，通常价值也比较低。或许，如果内部文档的作者能够看到，好的标题、作者和时间元数据可以改善搜索能力，那么情况将有所改善。

在任何情况下，许多常用文件格式类型受限于它们所能记录的元数据的类型。这导致了对记录文档细节的外部元数据资源库的依赖，或者，导致有些在检索中可能有用的元数据的缺失。

> **场景2**：一个机构使用Microsoft Word创建报告，用于外部Web发布。因此，为了使报告在被所有利益相关者阅读和打印时可以保持分页一致，他们将报告转换为PDF格式。不幸的是，员工很少会在Word文档属性中记录标题。Word将使用默认的文件名。当文档转换为PDF时，短语"Microsoft Word"被添加到文件名，并存储到PDF的元数据部分，然后被搜索引擎的PDF过滤器找到。当一个客户搜索网站的报告时，所有的结果都有相同模式的标题：Microsoft Word-fileXX.DOC。尽管它们实际上是PDF格式的，但很多搜索者会认为结果可能是近似重复的，并
> 651 且它们都是Word格式的。

第四个因素是，文本信息可能会只以扫描形式存在于文档中。很多办公室都配备了复印件/传真机/扫描仪设备，它们扫描打印的文档，将产生的PDF文档发送到指定的电子邮件地址。从PDF文件中提取文本需要使用OCR软件，这增加了时间和错误率。第五个因素涉及文档内的访问，它可能是以不同模式压缩的，也可能会加密。PDF格式文档可能在内部被标记，禁止文本提取。第六个因素，文档内部的重要结构可能被排版惯例规定的很多文档

类型所表示。例如：

> **场景 3**：一个机构要求员工简介必须按如下排版："由 12 磅 Times-bold 的姓氏开始，后面跟随着 12 磅 Times-Roman 的名。下一行包含就业类别，然后是服务年限……"

通过使用抓取（scraping）技术来在索引之前重建逻辑字段，检索系统在根据场景 3 创建的数据上的搜索性能有显著的提高。不同于 Web，有理由预期单一机构中只有少量这种类型的协议。

过滤大规模二进制格式文档可能是非常费时的。使用增量过滤，只需要过滤那些在上次更新后才改变的文档，所需的时间通常可以减少，

15.3.3 索引

企业搜索工具使用的索引格式没有理由与 Web 或其他地方不同。倒排索引（见[1798]）常常使用，通过额外的结构来表示文本注释（见 15.3.4 节）和静态得分。然而在企业中，有一个特别的需求——索引字段数据（将各种元数据类型，以及元数据类型和文档内容区分开来）。索引系统在对不同类型数据的支持度、索引数据的速率、高效支持短语和邻近操作的能力，以及所产生的索引的压缩度等方面表现不同。附录 A 比较了开源索引软件在这些方面的性能。

在设计索引时的一个挑战在于如何最好地处理增量式的内容更新。增量索引按如下方式处理更新：新的文档通过在文档表中增加新的条目，以及在它包含的每个索引项所对应的记录列表的最后增加条目来实现。这可能会需要很多随机存取的 I/O，并且可能会与为了支持高效的、以文档为单位的（Document-At-A-Time，DAAT [273，1046]）查询处理而进行的文档排序相冲突。在记录列表末端也可能没有空间来容纳新的条目，因此需要释放当前列表的空间，并在倒排索引的尾部创建一个新的列表。

从增量索引中删除文件会带来更多问题，特别是当记录列表压缩时。一个典型的解决方案是将记录留在原处，而将文档标记为删除。文档的更新可以视为先删除，后插入。随着时间推移，增量索引的规模会显著扩大，而访问速度会由于碎片和局部性的损失而下降。 652

使用增量索引的另一种方法是维护基本索引和更新索引的组合，并且并行地对它们搜索，如以下抽象出来的场景：

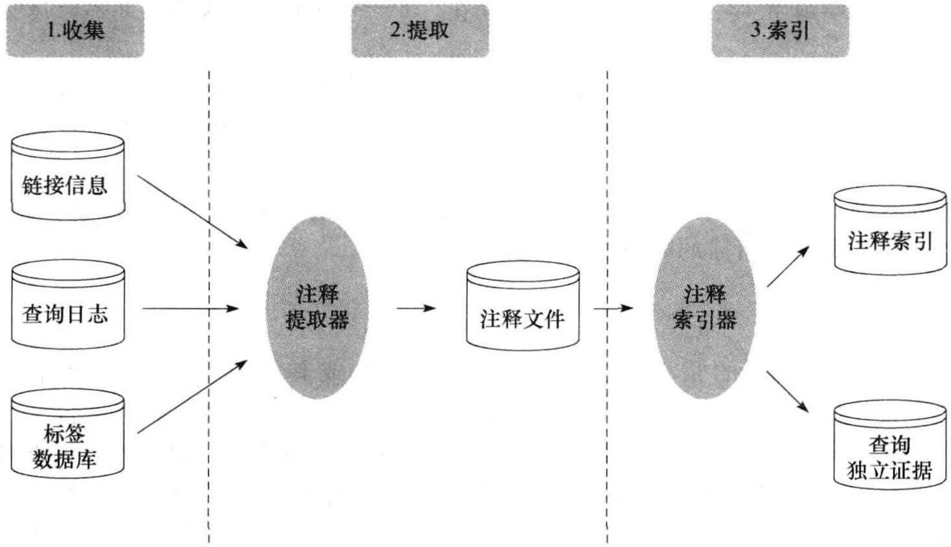
> **场景 4**：一个媒体机构拥有一个数百万个文档的网站。它在每个周末索引整个网站来建立基本索引。由于基本索引已经建立，所以每天晚上对所有在基本索引建立之后的新内容建立一个更新索引。将基本索引中更新的文档标记为"杀死"。每隔 20 分钟，对网站新闻部分的内容建索引。默认情况下，在网站上搜索，需要查询的元索引包含三个部分：基本索引、更新索引和新闻索引。

可以对这两种更新索引的方法进行很多变形。例如，可以建立一个超结构，使得基本索引和更新索引组合起来表现得像单个索引。另外，可以提供一个工具将基本索引和更新索引合并成为单一索引。

15.3.4 文本注释的索引

如同在 Web 上，企业文档（包括非 HTML 文档）可以使用不同的注释机制：通过链接的锚文本、正式使用的元数据、伴随点击的查询 [721，1743] 和分众分类法标签 [1131]。图 15-2 说明在查询处理中如何索引注释。它们可以用来提供查询依赖和查询独立的得分组

合。聚合注释可以分开评分，之后与文本得分合并 [1225]，或者可以将它们视为原始文档的字段 [1370]。



653 图 15-2 支持有效的企业搜索的注释过程

当然，机构在注释数据的类型和数量上有很大的不同。很少有机构支持分众分类法，但是有些机构用它们做实验。澳大利亚悉尼的 PowerHouse 博物馆[⊖]允许访问者到它们的外部网站来对展品的网页进行注释。如 [502] 所报告的，分众分类标签的实验已经在 IBM 进行，但数据量还很小。

15.3.5 查询处理

如 15.5 节指出，企业检索的查询常常因为查询语言与文档语言的不同而失败。企业检索系统可能提供同义词典扩展、查询建议、词干提取和相关反馈等工具来帮助弥补这一差距。

如在 Web 检索中，通过将文本产生的得分与静态得分相结合可以在企业内部得到更有效的排序。静态得分公式可能需要根据特定发布环境的特点调整。例如，在某些机构中，链接数和 URL 长度可能与答案是否有用不相关。而且，在企业内部，可能没有相互链接，且静态得分可能需要考虑新的因素，例如：

- 对资源的访问频率。
- 发布的新鲜度。
- 电子邮件的垃圾评分。
- 文件类别或流派（对于所有的查询，有些文档类别或流派往往比其他的更有用）。
- 资源库（相比于电子邮件归档中得到的结果，可能偏向于从员工数据库中得到的结果）。

在图书馆或者电子商务网站中，静态得分可能考虑某个项目被借出或购买的次数。此外，发布者的收益（例如项目的利润率、商品的易腐烂性和库存量）可能也在决定得分时被考虑。

⊖ http://www.powerhousemuseum.com/。

与查询独立的因素对于静态得分的相对价值可能在不同机构间有很大的不同：发布日期在很多机构中可能是可靠的，在另一些机构中则不是；如果 URL 是通过内容管理系统以固定格式（通常是晦涩难懂的）产生的，则 URL 特征的价值在机构中就严重缩水了；如果成千上万的链接指向同一个页面，这通常表示了它的重要性和流行性，但如果这些链接由一个人创建，从单一的导航模板产生，那么情况就不是这样了。

最优的搜索排序必须抑制近似重复结果的出现，鼓励多样性，以减少可能存在的 Office 文档的多个草稿和版本、销售资料的多种表示形式、在文件共享和电子邮件附件中的相同文档，以及多个 URL 发布相同资料的普遍问题。请注意，一般不宜在收集或索引中消除近似 654 重复。否则，限定信息资源子集的检索可能会失败，因为所需文件的唯一副本由于在其他范围有近似重复的文档而被删除了。

当试图提供“单一搜索框”来访问机构的所有信息资源时，异质性是排序困难的主要原因。对机构的 Web 文档进行打分可能使用一些因素，如 URL 结构和长度、链接数和锚文本。而电子邮件、员工数据库、CRM 系统或记录管理系统（Record Management System，RMS）则不存在这些因素。在这种情况下，是否可能自动地产生统一的排序，使得它在网页内容排序方面表现很好，并且对于其他内容没有很强的支持或者反对的偏见？也许不同类型的表示更合适。

企业范围内的检索系统的优化调整以及结果表示的最佳方式，在不同的个体和不同群体的员工间可能有很大差别。读者可以参考 15.6 节关于这个主题的讨论。

15.3.6 搜索结果的展示

在很多情况下，搜索结果的来源或类型可以帮助搜索用户决定这个结果有多有用。统一排序的结果中可能包括表示来源或文档类型的图标，或者表示产品和员工简介的缩略图。另外，搜索结果列表可能按照来源或类型分段。例如，来自员工目录的结果可能显示在来自企业网结果的上面，而企业网结果显示在外部 Web 结果的上面。常常会看到，在机构内部把搜索范围限制在综合索引内的一个明确定义的文档子集中。例如，人力资源部门的企业网搜索框可能只允许来自该网站的 URL 搜索结果。同样，机构可以提供其他搜索界面，用于单独搜索员工目录、政策和程序，或者员工公告。

在某些情况下，文档不是按照与查询的相关性的降序排序，或者向用户提供搜索排序选项，而是按照某个合理的顺序显示结果：电子邮件搜索结果可以按照日期顺序排序，最近的排在首位；出版物可以首先按照文件类型排序，然后按照标题的字典序排序。最后，对于一个旅馆查找门户来说，可能需要一定的随机性，来避免某个区域中的一两家旅馆有极大的商业优势，而其他旅馆相应变得极为不利。类似的考虑可能在一些专家发现应用中开始发挥作用。

企业搜索工具可以提供一系列额外的功能来帮助员工或网站访问者在搜索时得到最大的价值。这些功能包括：聚类、元数据分面统计（见图 15-3 中的地点、年份和格式分面，见[738] 中的聚类和分面的比较）、多文档摘要、拼写建议和关联查询。企业搜索系统可能也包括一些工具，对结果集进行深层次分析，提取常见的实体，如人名、地名和机构名、电子邮件地址、电话号码，或者输入查询的超短语等。搜索结果可显示在地图环境中（提供在被选结果附近搜索带标签文档的功能）。如同在 Web 上，对于图像和产品的企业搜索结果应该 655 包括缩略图功能。同样，对于视频片段的结果应该包括关键帧的缩略图。此外，用户可以注册兴趣轮廓来通过 RSS 或电子邮件激活自动提醒。

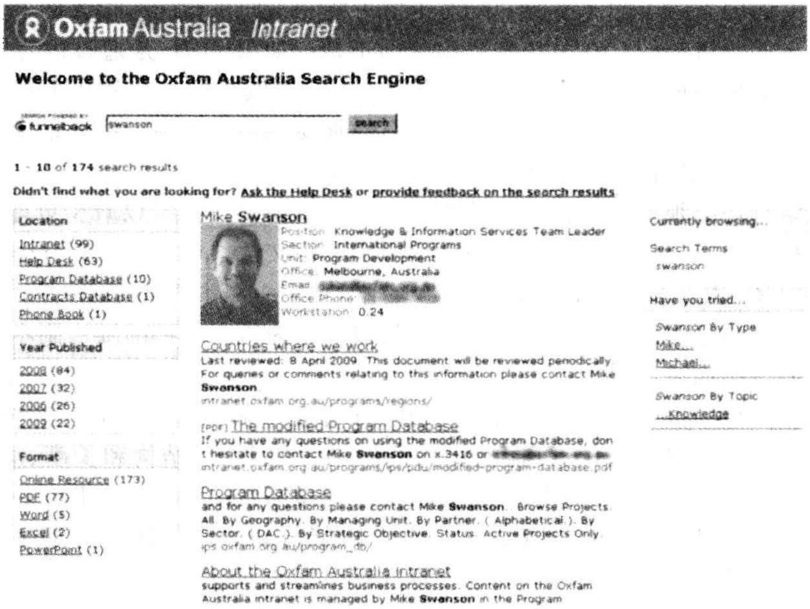


图 15-3 截图显示了在异质资源集上的企业搜索。注意员工电话簿结果中的图像、PDF指示符和截图左边的链接，它们允许搜索范围缩小到特定的资源库、特定的年份或特定的格式。还要注意到，引擎为用户提供了对获得的结果进行相关反馈的功能。Oxfam Australia 友善地许可了截图的使用

将搜索整合到企业应用中，并且在应用的环境中以相应的格式展示结果，通常是有道理的。例如，电子邮件搜索结果可以作为虚拟的电子邮件文件夹，支持所有常见的文件夹操作。另一个例子是，可以配置文档处理应用程序，使之不断地对最近添加到文档中的那部分文本进行搜索，并且所采用的搜索结果格式便于加入引用、摘引或者纳入正在处理的文档。

最后，企业搜索工具可以在由一些机构运营的定制员工界面中发挥关键的作用，如 Manber 等人 [1076] 介绍的 My Yahoo! 门户和 Yahoo! Companion 工具栏。门户网站页面可以按照个体需求定制，这样的想法在机构中和在 Web 上一样适用。定制的门户网页可以提供提醒、特定员工所需要了解事情的针对性摘要、活动链接和个性化搜索功能。在后台，搜索工具可以提供提醒和 RSS 源，当然还有量身定做、限定范围的搜索（如何对搜索进行个性化处理的细节见 15.6 节）。Manber 等人介绍了许多从应用于企业内部的 My Yahoo! 学习到的经验教训。他们强调用户界面设计的重要，并坚持界面行为必须是用户能预
656 测的。他们警告说，人们通常不理解定制化的概念，并且大多数用户不采用定制，所以应该投入很大的努力来优化界面的普遍版本。

15.3.7 安全模型

为了给授权用户提供全面的搜索，企业搜索工具必须有访问机构所有信息资源的万能权限。因此，工具必须百分之百地保证信息安全。由于安全模型的复杂性以及需要保持准确、高效的搜索，因此需要很多谨慎的工程化工作。查看文档或者搜索结果的权限取决于登录的用户（也包括搜索用户是否登录）。企业搜索工具的使用可能产生一些威胁，包括：

1）当内部的爬虫访问一个活跃的服务器页面或者 CGI 脚本时，会执行一些并非有意而为的动作。例如，一所大学的系统管理员给作者提供了 **delete-page.cgi** 功能，使他们可以通

过简单的浏览器请求来删除另一个网页。他们利用这个功能先删除网页，之后再进行编辑和修复。不幸的是，当他们访问 **delete-page. cgi** 时，在线 Web 服务器访问日志记录了这次访问，日志中包括指向所有访问过的页面的链接。当爬虫下次扫描网站时，它会发现 Web 日志页面，然后跟随其中的链接，导致那个新版本的网页被删除！

2）使人们能够通过搜索引擎看到那些禁止直接访问的内容。

3）妨碍人们通过搜索引擎来访问内容，虽然他们有直接访问的授权。

4）提供途径，使恶意的非授权用户能推断出敏感文档的存在，并可能推断出文档的内容。这方面最极端的例子是，虽然实际文档的链接被阻止了，但还是显示了搜索结果。不可访问内容的潜在信息来源包括命中数、分面数、多文档摘要、文档聚类，甚至响应时间。

5）外部可访问的企业网站搜索很可能受到跨站脚本、JavaScript 和其他类型的注入和缓冲区溢出的攻击。

对机构文档的访问通常受一些机制控制，如访问控制列表（ACL），可以指定一系列行为中的哪些（例如，读、写、索引）可以提供给特定的个人或组。文件夹或目录也受 ACL 控制。计算一个特定的用户是否可以访问一个特定的文档会很复杂，首先需要对用户身份进行检验，确定他们的成员组，以及根据父文件夹的 ACL 和文档本身的 ACL 链来检查他们个人与组的访问权限。机构可能会使用 Kerberos㊀等网络身份验证协议，并为企业实现一个单 657
点登录系统，避免用户对所使用的每个不同的企业应用都必须证明自己的身份。

从安全角度看，电子邮件和其他文档有些很有趣的不同点。电子邮件的作者（发件人）指定了接收者列表，但在大多数系统中不能为消息指定访问控制列表。相反地，消息的副本可能存储在接收者自己的文件系统的文件夹中，或者存储在中央邮件数据库中。消息的访问控制由接收者或者接收者邮件数据库的管理员所决定。因此邮件消息附件文档的最终访问权限可能与原来的很不一样。访问控制只可能在文件夹级别，而不是基于每条消息。

1. 文档集级的安全

在理想情况下（从简单性和效率的角度来看），机构的信息资源可以简单地分为拥有统一访问权限的多个文档集。例如：一般访问文档集、财务文档集、高级管理文档集、人力资源文档集。员工的搜索在符合它们角色的子集上处理，当提交结果时，不需要对每一个文档进行测试。不幸的是，在大多数情况下，适用的安全模型都比这更复杂。

2. 文档级的安全

当文档集级的安全模型不适合或者不能采用时，访问控制必须在文档级上应用。在后台，用户提交的查询在所有的文档上生成一个内部排序结果，然后必须过滤每个结果，排除所有用户不能访问的所有文档，以避免威胁 2 和威胁 3。为了解决威胁 4，其他所有搜索结果应该完全禁止。运营在安全环境中的搜索工具不应该显示当前用户无权查看的文档的结果片段（虽然简短），被排除的文档也不应该以任何统计或结果集分析的形式呈现给用户。

不同的机构对安全限制有不同的需求，需要在搜索时实施。在最极端的情况下，机构可能坚持当前的访问控制应该在搜索执行的那一刻在文档级实施，称为晚期绑定。如果在某天下午 5 点，员工的角色发生了改变，那么在当天下午 5 点零 1 分时，他们应该只可以看到所有符合他们新角色的文档。

Bailey 等人［127］说明了这个极端模型对于搜索响应时间的不利影响，这些影响由在

㊀ http://Web. mit. edu/Kerberos/。

大规模结果集中对每个文档检查访问权限而引起，特别是当原始文档存储在非本地的网络域
658 时。如果允许搜索引擎缓存所索引的文档的访问控制数据，那么就会减少一些延迟。在这个早期绑定模型中，角色转变的员工，在角色改变后的一段时期，继续访问的可能是符合他们之前角色的文档，要等一段时间才能对他们刚有资格查看的文档进行搜索。

15.3.8 联合/元搜索

当为机构内部的所有的信息源（库）建立一个统一的搜索时，企业搜索引擎对所有信息源都进行收集、提取和索引有时是不可行的。例如，从某些信息源进行收集处理是非常慢的，网络流量也非常昂贵或缓慢，或者数据量过大。这样的挑战也会在 Web 搜索中遇到 [99]。另外，数据可能被锁定在一个没有提供输出功能的专有应用程序中。（是的，真的!）如果有问题的信息源提供自己的搜索功能，那么仍然有可能提供一个统一的搜索，这个方法在不同情况下称为“搜索联合”、“元搜索”或“分布式信息检索”。对于联合搜索的更多细节在 10.7 节的分布式 Web 检索和 11.10.3 节的 Web 元搜索中介绍。

如果员工希望自己的统一搜索界面包括自己的个人（桌面）信息和机构外部资源，那么元搜索是唯一可行的解决方法 [1580]。在元搜索中，查询由代理程序接收，转发给联合资源的搜索界面。然后，代理程序将单独的结果结合成单一集合返回给用户。在这个领域的先驱工作是由 Gravano 等人 [667]、Voorhees 等人 [1653] 和 Callan 等人 [323] 进行的。由于不同搜索界面返回的排序和得分可能非常不兼容，因此产生了某些困难。一个信息源中排名最高的文档可能比另一个信息源中排在第 50 名的文档匹配得更差，而那个排在第 50 名的文档可能更面向主题。在最简单的情况下，这是由于某些项的 IDF 值（见第 3 章）在不同的信息源中可能有较大的不同。在更复杂的情况下，得分变化可能是由于不同的静态权重或者注释得分。

联合搜索对于维持文档级安全可能会带来某些特别的问题。用户凭据必须由代理程序转发给单一的搜索服务，只有依赖于这个过程才能正确地实施搜索。在所有要整合的信息源间提供可靠的单点登录机制几乎是不可或缺的。

元搜索的 5 个子问题

在元搜索应用中通常需要解决 5 个问题：

1）在服务定义时，识别和选择要进行联合的信息源。这可能是一本知名信息源列表的简单手册，或者它可能通过扫描来自动识别搜索界面 [424]。

2）在服务定义时，以及在服务操作期间尽可能经常地刻画信息源——它们索引多少文
659 档？被索引文档的语言模型是什么？使用的排序算法的效果如何？

3）在查询时，要选择包括在搜索中的可用信息源的子集。有些证据表明最优的信息源选择能够超过在所有信息源上进行的搜索，但是在实践中却很少或者从未完成。然而，信息源选择可以减少网络流量的花费、数据库订阅费用或每次查询的费用（这可能适用于特定的外部来源）。在一般情况下，选择依赖于在刻画过程中建立的信息源模型。

4）将查询翻译成被每一个联合信息源接受的查询语言。

5）在查询时，对每一个信息源中返回的搜索结果进行合并。

对于问题 2、问题 3 和问题 5，已经有大量研究在进行。问题 1 相对较少有人研究，因为要联合的信息源通常是给定的。问题 4 很少被关注，因为它通常是琐碎的或者难解决的——当搜索引擎在不同的语义模型上操作，或者支持不同的操作符集合时，准确的翻译是不可能的。例如，不可能把一个包含否定、合取和析取的布尔查询忠实地表现为词袋（bag

of words）形式。对于不支持截断操作符（通配符）或正则表达式的系统，也无法近似表示。

不幸的是，在大多数分布式信息检索的工作中，通过将 TREC 会议的随机检索数据划分为多个信息源的方式来进行模拟评价。这些划分在文档和文档集的类型和规模的变化方面比联合企业资源库要少得多，并且缺少文档类型或者人机交互数据等信息类别，而这些可能在企业联合中很有用。

需要联合的信息源可能以各种方式与代理程序合作。例如，它们可能提供关于文档集大小和文档频率的准确的统计信息。然而，在一般情况下，在企业或者个人元搜索应用中，要联合的信息源不会与代理合作。在不合作的情况下，就需要通过搜索界面从文档中抽样来刻画服务器。Callan 等人［321］提出了一种似乎十分随机的采样方法，但它的效果相当不错。接下来，Bar-Yossef 和 Gurevich［135］的工作采用了更加原则性的、基于双重拒绝采样的方法来避免采样偏置。为了表示在个人元搜索工具中哪些文档集可能会联合，Thomas 和 Hawking［1582］在一系列不同的文档集中评价了 4 种采样方法。Bar-Yossef 和 Gurevich 的随机游走方法发现好于其他可选方案，但是会带来相对高的成本。Thomas 和 Hawking 提出了一个更高效的*多重查询采样*方法，取得了类似的精度。

多重查询采样提交一些高召回率的查询，它们从文档集独立产生的数据池中采样而来，然后请求 k 个结果。根据搜索引擎的实际情况，k 尽可能高。它从结果集的并集中采样文档。未产生任何结果的查询（下溢）和那些产生多于 k 个结果的查询（溢出）被拒绝。使用结果集的并集往往有助于减轻*查询偏置*问题（很多查询倾向于返回那些含有丰富索引项的长文档）。拒绝溢出查询避免了*排序偏置*（静态分数高的文档被采样的概率会增加），而选择一个大的 k 值则能减少拒绝的可能性。 660

通过搜索界面来估计文档集的大小，通常依赖于那些为估算鱼类或动物种群而开发的方法。文档通过类似上面描述的方法来采样和重采样，而不是诱捕和释放动物。在简单的抓捕-再抓捕方法中，两个独立的、无偏置的采样中，相同文档的数量可以用来估计种群的个数。Shokouhi 等人［1473］描述了在这个领域的新方法。

现在已发布的关于信息源选择的方法已经超过 40 个！有些依赖于通过 STARTS 等探查协议［665］从信息源获得信息。其他方法假设项频数据可用，而另外一些则假设没有合作。Callan 等人提出的、著名的 CORI 方法［323］将每一个文档集看做一篇文档，将文档集的集合看做文档集，使用标准的相关度计算作为选择的基础。在不合作的情况下，其类似于 TF 和 IDF（见第 3 章）的信息必须从样本中估计。选择方法还可以考虑检索系统工作在每个信息源上的效率，因为如果它们不能及时响应查询，那么选择能得到好答案的服务器也会带来负面效果［437］。

如果没有提及 Fuhr［600］提出的决策理论框架，那么这一主题的讨论将是不完整的。这个理论使用检索相关和不相关文档的代价、期望的检索质量、每个信息源中相关文档的期望数量，以及文档传输和查询处理的开销，得到对每个信息源请求多少文档（可能没有）的最优决策。如前所述，对企业信息源联合环境中选择方法的研究比较匮乏。感觉在这种环境下，如果尝试将保存在特定资源库里的文档类型（如电子邮件消息、日历项、联系人详细信息、服务历史，技术手册等）与查询背后的任务相匹配，有可能找到更好的方法。

与选择一样，也出现了很多合并结果的方法。Lawrence 和 Giles［987］提出一个有效的合并方法，下载初步结果列表中的所有文档并在本地进行相关性排序。这个方法在 Web 上的缺点是对于每个查询产生的网络流量和完成结果列表合并的延迟。在企业中，这些问题

可能是可控的，但可能会出现其他的困难。例如，排序异质文档类型可能会造成困难，甚至更重要的是，专有应用程序可能无法输出完整的文档。Rasolofo 等人 [1335] 提出并评估了在当前新闻元搜索情况下合并结果的策略，其中信息源在所提供的每个搜索结果“片段”的类型和数量方面有很大的不同。他们能够获得来自于片段的结果，其效果接近于通过下载和本地索引完整文档而获得的效果。

元搜索结果的展示在异质环境中非常重要。不同类型的结果，如图像、客户联系信息和企业政策可能需要通过不同方式展示。此外，明确地告知每个结果的信息源也是有价值的。一种避免合并问题的方式是避免合并在一起，即对于不同的信息源，以单独的栏来展示结果
661 列表，就像几年前 A9[⊖] 多信息源搜索引擎的突出表现。另一种方法是在按信息源分段的列表中展示结果。尽管图 15-4 所示的结果列表分段实际上并不对应于单独的信息源，但它们可以这样做。另一个分段列表的例子出现在苏格兰护理监管委员会（Scottish Commission for the Regulation of Care），它们提供给检测员一个移动搜索界面，允许他们同时搜索多个数据库，并带有查询预测功能。当输入查询的前 3 个字母后，数据库中匹配这个前缀的结果就会显示。随着输入查询更多的字母，结果会逐步地完善。



图 15-4 分段结果列表的展示形式，这种设计方便不同类别的访问者访问网站。搜索用户可以点击链接来扩展某一特定类别的结果。这是来自国家处方服务部（National Prescribing Service）的截图

15.4 企业搜索评价

企业搜索评价可能是为了科学研究、产品测试或者公司的内部目的。

15.4.1 企业搜索的公开测试集

机构之间信息存储的巨大差异，以及大多数企业信息包含着机密，这些都使得很难建立
662 用来调试和比较搜索工具的测试集。有时，由于破产而使一个公司的全套数据可为公司外部

⊖ a9. com。

所用。安然（Enron）公司是一个众所周知的例子，虽然有些研究人员对于将它用于研究有一些道德上的顾虑，但是实际上很多研究还是基于 Enron 电子邮件语料库。

不幸的是，对完整数据的访问只部分解决了建立测试集的问题。此外，我们还需要了解，在公司正常的运营中会从事什么样的搜索类型，会提出什么样的查询，返回的结果会有怎样的价值。在一般情况下，可能很难去联系破产公司的前员工并且劝说他们分享信息需求和判断。

即使科学家被授权来复制一个机构所有的数据，并且对员工的搜索行为进行研究（他们搜索什么和他们选择哪个搜索结果），但是我们怎么可以确定这个机构可以代表其他的机构。我们怎样可以有信心，在这个公司数据上最好的搜索引擎在其他公司完全不同的数据上也会表现得一样好。

TREC 会议的企业搜索任务

据我们所知，对于企业搜索评价唯一公开可用的文档集（语料库＋查询＋判断）是 TREC 会议企业搜索任务建立的。由于前面提到的问题，企业搜索语料库仅包含在外部网站上公开的资料——w3c. org 的爬取结果、w3c. org 的邮件列表（转换为网页）、在 15.2 节中提到的 csiro. au 的爬取结果。有兴趣的读者可以参考在 TREC 会议网站[⊖]上发表的任务综述（例如 [126，439]）和第 4 章。

15.4.2 企业搜索内部评价

需要对企业搜索工具的效率进行实际评价的原因如下：

- 搜索引擎公司研发改善算法，并为排序函数中的系数选择好的默认值。这种研发需要在代表不同企业搜索环境的大规模测试集上进行。
- 产品的比较带来购买决策。
- 在某个特定的环境中，调试一个现有的系统来使它更好地工作。这种调试可能：
  - 通过提高公众查询比例来降低成本。这些查询可以通过 Web 来处理，而不需要昂贵的电话费用，或者面对面的支持。
  - 提高员工的生产力（例如：通过避免在重建已有信息上花费精力）和公司的竞争力。 663
  - 增加销量，通过确保潜在的客户可以很容易地找到产品和服务信息，并可以找到最便捷的购买方式（无论是在 Web 上或者通过传统的方式）。
  - 提高决策的质量。
  - 减少投诉。

企业搜索的评价和其他类型的搜索在原则上没有不同，但重要的是要确保评价忠实地体现真实的企业搜索。对同一个查询，将两个搜索功能的结果进行并排比较的方法（见 4.5.2 节和 [1581]）在这个环境中有很多优点。因为比较工具代替了通常使用的搜索工具：

1）如果所研究的用户组是使用搜索功能总体的一个均匀样本，那么关于总体推断的有效性可能只受限于无偏采样误差。采样误差当然可以通过增加样本大小而降低。

2）实验者没有必要了解（甚至知道）搜索用户在进行什么任务。对于每个搜索用户，要求记录的只是对每个搜索的投票（例如，“偏爱 A”、“偏爱 B”、“两者都没有用”、“两者

⊖ http://trec. nist. gov/。

同样好”)。

3）结果集被整体地评价。即使大部分文档是很相关的，评价者可能也不希望大量内容或意见相互重复的结果集。

4）提交查询的人知道他们为什么提交，并可以通过查看结果集是否满足其查询背后的需求方式来评估得到的结果集。实验者不需要决定：应该判断多少答案，需要采用多少级的相关性，应该采用什么测度，或者结果集内的重复应受到什么惩罚。相反，搜索用户下意识做的任何决策过程对于他们的任务都是适合的。

5）相比大多数人为的实验室检索实验，并排评价是真实的进行而不是模拟的环境。此外，为同一个人同时给出 A 和 B 两种条件，能够控制不同人的主观性差异，以及在不同时间点同一个人的判断差异。

有时，有人会担心并排比较缺乏敏感性。尽管 TREC 会议的随机检索评价可能发现有百分之几的 MAP 差距，但是，即使在数十个搜索用户评价足够数量的查询后，对于系统 A 和系统 B 间仍然可能没有明显的偏好。不过，这真是这种方法的缺点吗？如果系统 A 正在大量使用中，并排研究清楚地表明，用系统 B 代替系统 A 得到的好处是很小的。用户投诉量不可能减少。

应用并排工具进行 n 路比较，当 $n=2$ 时达到最优。按照目前的显示技术，有可能使 n
664 超过 2，但是判别结果的次数需要增加，并且随着判别开销的增加可能会干扰到搜索者的工作。在 [941] 中，$n=3$，要求实验者在同一尺度上对每个面板定级，而不是表达对每一对的偏好。

并排方法的一个更大的限制是无法使用它进行调整。一个典型的企业排序函数需要结合 20 个或者更多的变量。调试组合函数需要大量的偏好数据，而它们没办法通过并排比较得到。如同在 Web 上，对于点击数据的分析可以用于挖掘偏好关系，见 4.5.5 节。

15.4.3 企业搜索调试

为了调试企业搜索系统，人们可以使用一个传统的，但公司私有的测试集或者采用机器学习方法来大量收集这种形式的数据：“对于查询 Q，文档 D_1 明显优于文档 D_2”（基于控制偏置形式后的点击频率）[841，844]。以 TREC 随机检索的风格对测试集进行判断，可以由机构的员工手动建立或者依靠用户的点击数据。

在评价中依靠点击数据会面临一些风险。第一，它可能会受系统性偏置的影响，更偏向于文档的标题、URL 和结果集片段，当标题、URL 和结果集片段相关而文档不相关时，往往将文档作为相关的，反之亦然。第二，搜索引擎排序函数可能以各种方式使用用户点击数据，包括查询依赖的和查询独立的。用一个基于点击频率的代价函数调整排序是有风险的，往往会高估函数的点击成分，导致可用点击数据很少的，或者点击数据有误导性的查询，其表现比应有的差。

当使用测试集进行调试时，文档集（文档、信息需求、判断和测度）应准确地代表实际情况；否则，从测试集上决定的最优参数设置可能离实际应用的最优值相差很远。

企业搜索的工作量忠实地记录在搜索引擎保存的查询日志中。一个明显的无偏评价方法是从查询日志中均匀地随机抽样，并试图找出对于查询背后的信息需求最有用的答案。[1389] 表明对搜索引擎所预测的性能水平有可能会有很大的变化，这取决于查询集如何选择。那些可选搜索引擎的排序可以很容易地改变，取决于所选择的用于评估的那些查询。

工作量抽样方法的限制在于，需要从日志中的查询推断出信息需求。在某些情况下，解释是很明显的，例如“薪资标准”和“知识产权政策”，但是在另一些情况下，提交特定的查询可能有多个原因。在有些情况下，查询的含义完全是神秘的，或者查询是在范围之外，即在文档集中没有可识别的、有用的答案。

另一个问题是企业搜索引擎（特别是当用来提供外部网站搜索时）是为发布者的利益运营的，而不是为了信息消费者。在很多情况下，发布者的利益和网站访问者重合；但在某些 665
情况下，它们不是。发布者可能关注于对某些查询的评估，这些查询与机构的关键业务有关。例如，银行可能要求其网站搜索对查询“抵押”、“房屋贷款”或“信用卡申请”提供完美的答案集，但是对于过去的年度报告或者收费表的请求就不那么有兴趣。许多机构都希望搜索引擎的性能有所偏向，以便达到某些商业或政治目标。

由 CSIRO 发布的 C-TEST 开源工具[⊖] [722] 为测试文件提供了一种正式的表示方法，它可以对很多因素建模，这些因素对于有意义且可重用的企业搜索评价是必要的：

1）给测试文件中的每个查询都关联一个权重以反映重要性，它可以由商业因素或者提交的频率决定。

2）对同一个查询可以显示多个解释。

3）文档集中的重复文档可表示为和单一文档具有同等价值。这可以防止搜索系统通过对相同的文档返回多个副本或版本来提高得分。

4）相关的答案可以设置不同的权重以反映它们对于满足查询的特定解释之后的信息需求的贡献。

5）给定查询之后的需求，对该查询的测试文件条目能够指定合适的判断深度。对于情报评价或者科学研究来说，对排在前 1000 篇的文档进行判断可能是合理的。然而，当在企业网上寻找人力资源部门的主页时，员工不可能会耐心地查看前 10 个之外的结果。

在写本书时，开源搜索系统 Lemur、Terrier 和 Zettair 的维护者都同意为 C-TEST 格式提供支持。

15.4.4 所能期待的是什么

我们所能期待的企业搜索工具的性能在下面极端情况组成的连续区间中的某处：最可能的答案排在第一，以及在许多不相关的答案中点缀着一些不完整的答案，正如我们现在讨论的。

1. 最可能的答案排在第一

如果我们在当前的 Web 搜索引擎上查询一个公司的名称，如“福特汽车公司”，或者数学概念的名字，如“强连通分支”，那么很有可能这个公司的最佳答案页面（它的主页）或者概念的权威定义会出现在搜索结果的第一位。这种成功依赖于丰富的 Web 搜索环境：链 666
接图、锚文本、URL 的长度和结构、用户行为数据等。它还依赖于为回答这些需求而专门发布的信息的可用性——公司网站、维基百科或其他提供高质量定义和解释的网站。

2. 在很多不相关结果中点缀着一些不完整的答案

TREC 会议随机检索评价竞赛建模的任务（1992—1999 年）（见第 4 章和 [1654]）本质上是从新闻档案中收集信息。当用最先进的搜索工具在非常小的文档集（即 50 万篇文章）上处理 TREC 的随机查询时，如“轮胎回收的经济影响”和“前苏联的裂变材料蔓延所带来的影响”，得到的答案远远无法令人满意。即使是最强的 TREC 参与系统也未能找到一半

⊖ http://es.csiro.au/C-TEST/。

的相关文档，并且在前 10 篇中返回了 5 篇或者更多的不相关文档。在这种搜索中，没有像 www.ford.com 这样的明确答案，而一篇文章比另一篇文章更重要的标志也难以辨别。没有链接结构，没有锚文本，没有站点结构，在 TREC 的随机搜索任务中也没有用户行为数据。而且，文档不是为了满足这些查询的信息需求而存在的。此外，查询可能与它应该匹配的文档使用不相同的词：例如，“前苏联国家”应该匹配“俄罗斯”、“乌克兰”等（也可能是相同索引项的西里尔字母等价表示），而“裂变材料”应该匹配“U-235”、“钚”等。在这种搜索环境中，通过更好地执行文本操作，好的搜索引擎能与较差的搜索引擎区分开来，这些操作包括：查询扩展（词干提取、英美拼写合并、伪相关反馈、同义词典）、文档长度归一化和相关段落权重提高。

3. 企业搜索在连续区间的哪个位置

在一个精心组织的、如同 Web 缩影的企业网中，对于部门、员工和服务的搜索可能会在区间愉快的那一端。另一方面，如果被搜索的信息包括数据库中的纯文本块，或由 Office 文档转储为没有元数据或命名约定的非结构化共享文件，那么基本性能会降低，用户会不满意。对后一种情景，搜索效果顾问可能找出将报告的当前版本与草稿区分开的方法，并且找出各种查询独立的因素来用于改进排序。同时，他们还可能建议简单地改进信息发布和存储的方式，作为改进搜索效果的手段。

15.5 不满意的可能原因

前面已经指出，员工对企业搜索的满意度和访问者对网站搜索的满意度通常很低。满意
667 度取决于“搜索和可搜索性”，即搜索技术的效果，和有多少有效的信息和服务发布。有时，最好的答案没有匹配查询，也许由于语言（例如，查询是“门”，文档讨论的是“手工操作的个人出口机制”[⊖]），也许因为查询词只以图形的形式存在（例如，扫描的文档）。在这种情况下，似乎更好的是改善信息发布的方式，而不是试图修改搜索技术。

据我们所知，当前的搜索技术都是基于统计的。在这些系统中，给定查询，期望的文档检索得分取决于查询与文档（及其注释）的匹配程度和反映文档有用的可能性的静态得分（如 Web 搜索）。见 15.3.5 节中有关企业搜索静态得分的可能成分。一篇文档的排序自然取决于其他文档所取得的得分。考虑信息在特定网站上发布的方式，从而调整排序算法，能够使效果有很大不同。

表 15-1 列出了为什么对于特定的查询，所期望的文档排名不高的几个原因。但是，匹配和排序只是企业搜索面对的一部分问题。在关于企业搜索工具的抱怨中，有很高的比例实际上是因为需要的文档甚至不在搜索引擎的索引中！表 15-2 列出了出现这种情况的一些原因。值得注意的是，通过使用非常特定的查询，应该能够确定所需要的文档实际是否在索引中（可见）。例如，将文档标题作为短语查询，或者搜索在文档 URL 中的词语。

表 15-1 一篇关键文档在索引中。为什么对于这个查询它的排名不是第一

	主要诊断	特定问题	解释
1	文档真的与查询匹配吗	• 所有查询词都出现在文档的索引文本中吗 • 潜在地依赖于词干提取、同义词典扩展、部分匹配、拼写校正、语义理解、语言翻译，或读心术吗	或许搜索工具不能支持这些事情

⊖ 感谢《Private Eye》提供的例子。

（续）

	主要诊断	特定问题	解释
2	该文档不如其他文档与查询匹配得好吗	• 文档中包含的每个查询词的个数有多少 • 查询词是否相邻出现，尤其是作为一个短语 • 查询词是否较早出现在文档中 • 是否（所有）查询词都出现在文档标题或主题词中（作为一个短语）	很多搜索引擎根据隐含在问题中的特征来分配得分。你可能需要查看文档源以确定查询词没有出现在 NOINDEX（非索引）部分
3	较短的文档也可以同样匹配查询吗		很多排序算法的匹配分数用文档长度进行归一化
4	排名更高的文档有更多的入链或匹配查询的文本注释吗	• 它们有很多来自其他网页的链接，并在锚文本中使用查询词吗 • 它们被与查询匹配的分众分类法标签标记了很多次吗 • 当查询提交时，是否很多用户点击排名更高的文档	这些特征在提高 Web 搜索质量方面是有效的，也能够在企业网上获得成功
5	排名较高的文档具有表明它们更流行或重要的特征吗	• 有更多入链吗 • 有更多分众分类标签吗 • 有更多用户点击吗 • 有更短或更简单的 URL 吗 • 最近有更新吗	有些搜索引擎提供工具来允许显示这些因素
6	排名较高的文档来自与期望的结果不同的资源库吗	• 排序函数“意外”喜爱来自特定资源库的结果吗	管理员可能配置系统使得来自一个资源库的结果优于另一个，或者鼓励资源库的多样性 668~669
7	目标文档与出现在排序中的其他文档非常相似吗		想要的文档可能已经作为近似重复的文档被删除或者在排序中向下压

表 15-2 为什么所需要的文档似乎不在搜索工具的索引中呢

	诊断	解释	可能的修正
1	此文档存在吗	令人惊奇的是，这是抱怨的一个真实原因。可能应该有一个关于 X 的报告，但是它实际上不存在	创建缺失内容吗
2	它在范围内吗	这是一个很常见的失败原因。期望的文档位置（例如，外部网站）不包含在搜索范围内	改变默认范围来增加可能性。确保所实施的范围限制是经过说明的
3	允许我看到它吗	典型的是，限制某些员工看到太多的文档。你登录了吗？公司可能不希望你看到这些文档	如果适合，开始改变权限
4	采集者可以访问吗	如果一个企业网或者网站的文档无法被链接，那么它可能没有被爬取或索引。使用数据库和文件共享，配置错误可能也会导致丢失内容	保证 Web 内容可以链接到。检查允许和排除的模式。检查如 robots.txt 的机制和机器人元标签
5	如果这个文档是二进制的或专有格式，如 JPEG、PDF 或 MS Word，那么它的文本内容可以被提取吗	• 对这种内容的过滤器安装了吗 • 如果是 PDF，允许文本提取吗 • 如果文本内容是用图形表示的而不是文本形式的，那么系统可以通过 OCR 识别图像吗？如果可以，OCR 软件会曲解查询词吗	以容易理解的格式发布。确保安装必要的过滤器
6	上次更新索引时，文档存在吗	收集、过滤和索引操作可能间歇地工作（例如，每周），而不是连续地工作。在上次收集操作后发布的文档不在当前的索引中	检查收集日志。改正错误。发起重收集
7	文档被标志为禁止显示吗	• 管理员已经禁止了发布吗 • 文档包含 NOINDEX 标签或注释吗 • 文档过期或者超过了合法期限了吗	改正发布的错误 670

有一些具体的匹配问题，这些可能导致企业搜索中的排序问题。在很多大学中，地图是一个经常提交的查询。经常有一些匹配页面拥有很多入链和高度匹配的锚文本，但它们不是这个查询的好答案。这些是大学范围内很多网站发布的“网站地图”。类似地，对于查询校长（或院长），应该在副校长（副院长）的页面之前检索到校长（院长）的页面。最后，对工程学士的查询，应该首先检索到这个特定学位的信息，其次才是组合学位，如工程和商业学士。

有很多方式能解决这些匹配问题。如何找到它们将作为一个练习留给读者。

15.6 情境化和个性化

除了索引更新之外，无论何时，无论谁提出的，一个简单的搜索引擎对同一个查询会提供相同的结果，并以同样的方式表示它们。但在现实中，不是所有的用户都是相同的，如果可以得到并利用下面这些问题的答案，那么搜索性能可能会得到改进：谁正在搜索？他们正扮演什么角色？他们感兴趣的是什么？他们为什么搜索？他们在何处？他们正在执行什么任务？他们已经知道什么？他们能够理解什么？

个性化信息检索只代表个性化研究广阔领域的一方面。Pierrakos 等人 [1262] 调查了这个广阔的领域，并概述了个性化系统的一些可能的功能。一个完全个性化的企业搜索系统可以提供个性化的门户布局，它有用户特定的信息显示、可定制的外观和感觉、有针对性的提醒、个人搜索历史记录、可定制的搜索范围，以及与当前讨论相关的、偏向个人需求的搜索结果。

众所周知，如果情境是已知的并可以利用，那么个人搜索就可以得到改进。Teevan 等人 [1571] 为了量化个性化搜索的潜力，要求 15 个人在给定的明确搜索意图下，为一个查询所返回的 50 个网页按非常相关、相关和不相关分级打分。对于实验者提供的查询，他们发现了查询意图的多样性。即使意图是相同的，他们对结果的打分也有很大的不同。

Pitkow 等人 [1275] 在一个实验中展示了真实的搜索效果收益。48 个 Web 用户被分成新手和有经验的用户两个组。他们研究了在用户和 Web 搜索引擎间插入个性化客户端系统 Outride 的价值。Outride 是浏览器附加组件，它建立了一个基于用户的搜索和浏览历史，以及人口统计学和应用程序使用轮廓的模型。它扩充了用户提交的查询，并用以下方法处理来自后端搜索引擎的很大规模的结果集：结果分成“见过”和“没见过”两类，根据向量空间形式的用户轮廓来重新排序。Pitkow 等人观察到，完成搜索任务的时间，以及鼠标点击或者键盘输入等用户行为，都惊人地减少了。

搜索工具的行为可以根据组别、个人或任务的特性来定制。在讨论“基于用途”的信息检索方法中，Pitkow 等人 [1275] 指出检索系统能够工作在不同颗粒度的用途数据上，并且需要的话可以回落到较粗的级别。对机构内员工的情境化搜索有特别的潜力，因为对个人可以了解得更多，包括他们在机构内的角色和他们可能会执行的任务。然而，我坚信应该始终提供一个“基本配置（plain vanilla）”选项，并且搜索用户始终能够发现关于他们搜索特点的假设。

在讨论情境化搜索主题时，我们首先回顾搜索引擎对查询返回的结果集进行情境化时可用的控制和工具，它们可以用来改进排序（假设将对结果进行排序），对于特定需求优化结果的展示。接下来，我们讨论客户端和服务器情境化的问题。我们下面会探讨搜索情境向量可能存在的高维度，怎样才能通过从搜索设置中派生用户轮廓来降低维度，怎样才能确定它的值，以及怎样才能把它们传递给搜索服务器。

15.6.1 情境化的控制和工具

假设我们知道一些关于搜索请求情境的有用信息，这些情境信息可以通过什么方式影响搜索系统的行为呢？总之，搜索引擎控制有五类：范围、静态排序、查询操作、动态排序和展示。对于一个特定个人或组别，这些控制的设置可能记录在*搜索轮廓*（search profile）中。我们会在本节的后半部分讨论轮廓的不同类别和定义它们的方式。很可能存在一个完整的全局轮廓，它有效地定义了基本配置。按照 Pitkow 等人 [1275] 所描述的方式，这个全局轮廓可能在特定的搜索中被相应组别、个体和任务的局部轮廓所覆盖。 671

1. 范围

搜索范围是用来与查询匹配，并允许展示给用户的一个文档的完整集合。范围受到加入搜索过程的资源库的控制，受到用来在那些资源库中匹配文档的任何排除过滤器的控制，也受到特定搜索上的访问限制的控制。很容易看到，范围对于情境化搜索是一个强大的工具。例如，如果客户关系管理和财务资源库没有包含在搜索中，那么研发部门的技术人员对于搜索结果可能会更满意。

排除过滤器可以根据文件类型、媒体类型、流派、阅读年龄、日期、URL 模式或元数据特征从包含的资源库中排除一些文档。我们已经讨论过一种个性化范围：文档从结果列表中过滤，因为提交搜索请求的用户没有得到访问它们的权限。另一个与 Web 类似的例子是成人内容过滤，其中个人的偏好信息（或者他们父母的）用于规定搜索结果的范围。尽管成人内容在大多数企业或许是少见的，但类似的过滤技术可以用来为某些个人或组别抑制某些类型的文档（如技术手册）或者某些企业子网（例如，员工幼儿园或男员工的曲棍球队）。

个人元搜索 [1580，1583] 是范围的一个特定例子，其中个人选择随着时间推移可能会对他们很重要的资源，见图 15-5。对于一个特定的查询，个人元搜索系统可能会根据信息源特点以及个人过去的行为，选择信息源集合的一个子集。

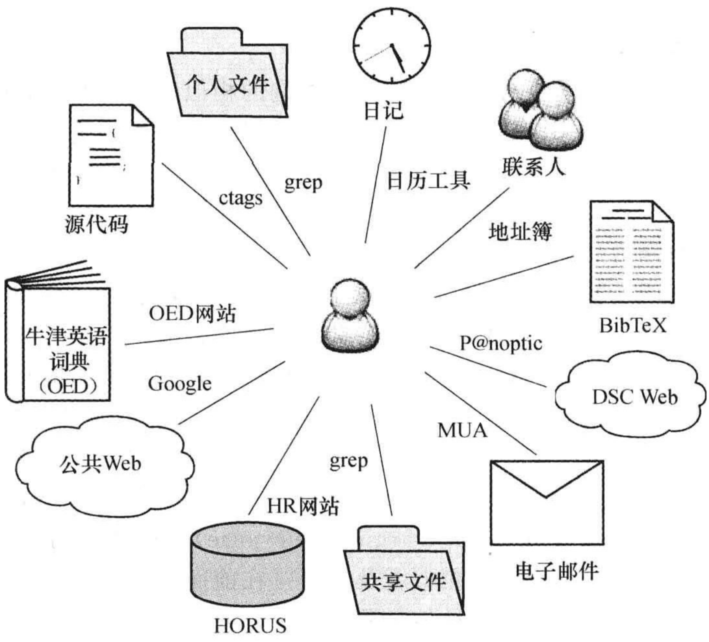


图 15-5 一个特定的个人元搜索配置，显示了统一搜索中包括的一系列信息源。注意在个人、工作组、企业和外部层次都存在着信息源。由 Paul Thomas 授权提供 [1581]

2. 偏置

与通过范围完全排除一个特定内容类别不同，排序算法的偏置可能更有效。正如我们已经看到的，现代企业搜索引擎在它们的排序函数中包括一个静态成分，它是很多查询独立特征的加权组合，如作者流行度得分（例如，HITS、OPIC 或 PageRank）、用户流行度得分（例如，由点击派生的测度）、文档新旧程度等。Jeh 和 Widom [831] 描述了一个系统，它在一个全局的 PageRank 向量中加入了个性化的部分向量，为网页的重要性和流行度的个性化视角提供一个可扩展的实现。这个方案与一些企业 Web 相关，但是这种企业网的规模要
672 小很多，小到至少对于机构内的用户组别，它可能可以保持完整的个性化向量。情境化用户流行度数据的一个明显形式是，记录基于个人或组别的用户交互数据，并且使用特定个人或用户组的流行度得分。

所有可以使用在范围上的特征也可应用在偏置搜索结果上来支持或者反对这些特征。例如，排序函数可能只是偏向于反对技术手册，而不是完全排除它们。如果其他类型的相关文档很少，那么技术手册可以被检出。而且，总经理撰写的文档的权重可能被提高，也可能被降低。总之，搜索系统的行为可以通过使用依赖于情境的查询独立的权重向量来进行情境化。

3. 操作查询

用户提交的查询可能以不同的方式进行操作，以便进行结果的情境化。在简单的情况下，组别特有的（group-specific）同义词典可能被用来以最适合该组的方式解释有歧义的查询项。例如，查询 CRM 在销售部门可能被认为是*客户关系管理*（customer relationship management），而在生产部门被认为是*碳纤维增强成型*（carbon reinforced mouldings）。

通过使用伪相关反馈技术的变种进行查询扩展，可以得到更高程度的自动化和成熟性，
673 如 Teevan 等人 [1570] 所描述的。他们使用伪相关反馈对 Web 搜索查询进行扩展，从而搜索用户个人计算机中的文档。从 Web 搜索引擎得到一个比通常更大的结果集，通过使用扩展的查询对文档进行本地重打分，将这些结果集重新排序。Teevan 等人观察到，相对于基准排序，该方法有小的，但统计显著的改进。

4. 操作动态排序函数

如在第 4 章中所见，几乎所有对文档内容和查询的相似度打分的函数都是参数化的。所以，尽管还不是很清楚为什么特定的参数设置可以更好地满足某些用户，但有潜力通过调整参数来针对特定用户或用户组改善结果质量。通过调整文本注释的使用，也许有更多的余地来满足特定组的利益。例如，只查看与某人相似的其他人所使用的分众分类法标签，或者只考虑由某人所属的组别提交的伴随点击的查询。

特定的语言特征或转换可能会更好地满足某些用户（或任务）。例如，对于词干提取，人们可能倾向于重度的、轻度的、没有，或者针对特定语言的。当执行某些任务时，自动地将美国和英国的英文拼写合并，对于搜索引擎是个优势。有些人可能偏好于包含非重音字母的查询应该匹配相同词的重音版本，例如“canon”应该匹配“canon”和“cañon”，但另一些人则不是这样。

有些排序函数包含一些用于改善返回结果多样性的组件。例如，Carbonell 和 Goldstein [332] 使用*最大边际相似度*（maximal marginal relevance）。在他们的方案中，一个文档在最终排序中的位置取决于它与查询的相似度及它与排在前面文档的向量空间中心的不相似度的组合。其他的方案试图包括来自多种信息源的结果，或者多种结果类型。在某些检索系统中，多样性的定义和所得到的回报按个体或组进行配置，但是我们现在还不能列举出在实践中用来提升效果的例子。

5. 人机界面的定制

按个人或组别的需求，定制搜索结果展示的方式有很多种。你想要看到排序列表形式的结果，还是缩略图网格？你喜欢搜索页面上列出简明的结果，还是较详细的？你偏爱哪些颜色或字体？你最喜欢什么语言？如果显示的是所有答案的摘要，而不是单个答案的列表或网格，你会选择它吗？你喜欢看到分面统计、查询建议和其他附加的服务吗？你喜欢在搜索的顶部有突出显示的与查询相关的新闻条目吗？你偏爱或需要将搜索结果读给你，而不是显示在屏幕上吗？

上面的例子适用于结果的展示，但定制也可能在输入方面。你更偏爱于输入你的查询，或者说出来，还是写下来？你喜欢系统协助你自动完成查询输入吗？——根据自己的查询历史，或者根据你所在组别的分享历史？很多这些问题涉及辅助功能，它们在企业背景中非常关键。如果一个残疾的员工不能使用机构的搜索工具，或者使用有些困难，那么他们可能很难有高的产出。 674

最后，员工可能通过电话或屏幕面积有限的移动设备来访问企业搜索引擎。在理想情况下，这些设备的局限性和能力能够通过搜索工具与合理构造的交互而识别。在 Web 中，装有 GPS 系统的移动设备，或者知道用户工作站的实际 IP 地址可以用于定位，然后根据用户所在的国家或地区来定制搜索结果（和交互类型）。有些跨国公司可能在内部使用这些功能，但是绝大多数公司因为太小或者过于本地化而没有这样做。

15.6.2　情境化：本地、企业或全球

之前对情境化的大部分讨论在搜索定制发生的地点上故意含糊其辞。通常，企业中那些承担丰富知识和信息角色的员工，分配有个人计算机供他们使用。有时，个人计算机以集中化的方式工作，可以从集中的企业或工作组服务器访问软件，文件和电子邮件也存放在那里。然而，在很多情况下，个人计算机必须以独立的方式工作，或者因为公司的 IT 政策，或者因为它是一台可在家、在客户现场或者在会议上使用的笔记本电脑。无论具体的安排是什么，个人计算机总是能够收集到大量的个人交互信息——用户接收、下载、查看、归档、发送、编辑、打印、收藏和搜索了什么。个人计算机也可以潜在地监控外部活动——人们对什么样的社交网站进行访问和交互，他们发送或接收了哪些“tweets[⊖]”和即时消息。

几乎所有对个性化有用的信息主要在个人计算机上收集或可收集，虽然它们中的一些也可由机构中的服务器收集，更少量的一些可由机构外部的搜索和其他服务收集（例如 ISP）。这个归纳的一个小的、但正在不断增加的例外是，大量的交流和信息交互现在往往出现在移动设备上。有些主管通过黑莓和 iPhone 来处理他们大部分的电子邮件。然而，电子邮件和联系人通常与他们的个人计算机同步。

情境化的一个重要的元素是正在执行任务的性质（见 15.2 节的一些例子）。在企业内部利用这点有很大潜力，因为能够从现在正进行的应用准确地推断出任务。例如，如果一个员工运行公司的项目成本估算应用程序时，同时执行搜索，那么很合理地推断他们的搜索是在估算项目成本的情境下进行的。如果搜索功能被嵌入到应用中，那么这种推断将会更加可靠。 675

客户端还是服务器

如上所述，对个性化和其他形式的情境化有用的大多数信息通常可以在个人计算机上得到。客户端个人计算机很显然是保存搜索的个人和任务轮廓的最佳场所。如果轮廓只保存在安全的个人计算机上，那么上面讨论的隐私风险便可以得到控制。不幸的是，一部分在外部搜索引擎上（如 Web 搜索引擎）进行的个性化搜索任务在搜索引擎上工作得最好，而不是在客户

⊖ http://twitter.com。

端个人计算机上。在外部搜索引擎，对于范围、静态得分和排序的修改可以充分地发挥作用。

在关于用 Web 用途挖掘（Web Usage Mining）帮助个性化的综述中，Pierrakos 等人[1262] 概述了一系列可以用来收集 Web 用途数据的方法，包括客户端机器上的日志工具栏、数据包嗅探器、保存在客户端和服务器间各个点上的 Web 日志和服务器日志。他们还概述了一些问题，包括在远程收集的数据中可靠地识别个人和会话的边界。

Web 搜索引擎为个人保存轮廓信息，以 cookie 的方式按查询先后顺序维护。然而，这样保存的个人轮廓是不完整的，特别是，如果个人使用多个搜索引擎，并且不与搜索引擎共享电子邮件或文档时。轮廓也只是描述了被特定搜索引擎所支持的个性化。另一方面，机构内运营的搜索工具，包括个人元搜索，提供了更多潜在的自定义的范围，隐私的问题更容易管理。

当然，可以与外部搜索引擎交流交互历史、轮廓或部分轮廓，但是我们接着就会被一些问题所困扰：我们保留隐私的愿望遇到困难；我们必须用适当的方式与搜索引擎交流轮廓中有用的部分，以便既能达到想要的效果，又不会产生过多的网络流量或服务器负载。

15.6.3 轮廓的隐私

有很好的理由来保留交互历史和个人搜索轮廓的隐私。这些资料可以向竞争对手提供非常有价值的信息，它们对定向广告来说是非常有用的数据，在比较少见的某些场景下，还可能是离婚案律师、勒索者、警察和外国情报机构强有力的资料。不要忘记，我们正在讨论的某些轮廓属于那些利益和活动比普通人更重要得多的人物。任何关于富有的企业收购者或美联储高管所执行的搜索和阅读的文档的信息都可能会引起市场投机者极大的兴趣。另一些人可能会在国家总统或缉毒局长的搜索轮廓中发现更大的价值。

1997 年，一个关于开放轮廓标准（Open Profiling Standard）[⊖] 的提案提交给万维网联
676 盟，这将保证轮廓信息的安全交流，但是它尚未得到采用。

15.6.4 定义、建立和维护轮廓

有些研究人员，他们对搜索行为的所有情境都感兴趣，而且会把所有的“生活点滴”（my life bit[⊜]）以及所有对气象、地理、心理和社会因素的描述都认为是相关的情境。换句话说，他们的目标是理解人们和他们的在线行为，而不是提高单个搜索“事务”的价值。

如果我们考虑更一般和实际的情况，那么很难准确地说明应该记录什么信息，以便得到上面提到的、对于某个特定人的特定搜索的最优控制和工具设置。上面提到的、可能用于产生搜索轮廓的交互信息，可以通过运行在本地操作系统层次的软件或浏览器和其他个人或企业应用的附加组件或插件记录下来。很容易完整地记录所有的交互事件（和对象）和它们发生的时间 [17，1570]。其他较不完整的记录可以保存在离用户较远的地方，如代理服务器或搜索引擎等应用上。

另一个要回答的问题是，如何确定哪些轮廓应该应用于搜索。同一个人可能喜欢不同的轮廓（或一个也没有），这取决于他们在某一特定时间内从事的活动。我们是否能够建立一个自动确定正确轮廓、正确率达到 100%的系统？可能不会。但是，我们怎么能在不使用户混淆的情况下，向用户解释哪些轮廓是可用的以及它们如何不同。在一个企业中，最好的方法有时可能是让用户从一个命名清楚的轮廓组中进行选择（如果他们愿意的话），如销售、

⊖ http://www.w3.org/TR/NOTE-OPS-FrameWork。

⊜ http://research.microsoft.com/en-us/projects/mylifebits/。

财务、人力资源、研发和基本配置。

15.6.5 用户建模

自动产生轮廓的过程可能描述为*用户建模*。现在已经有一些不同种类的用户模型。

1. 本体向量

Pretschner 和 Gauch [1302] 描述了一个系统，其中用户轮廓由一个带权向量组成，向量每一维对应在某个公开本体中 4400 个层次类别中的一个。每个类别用一个文档向量表示，它代表这个类别的 10 篇典型文档的混合。当用户访问一个网页时，会计算这个网页相对于每个类别的相似度，并且轮廓的权重就由一个关于这些相似度、页面浏览时间和页面长度的函数进行更新。研究发现，随着时间的推移，轮廓会收敛，并且轮廓可以用来对来自后端搜索系统的结果进行重排序和过滤。通过重排序，11 点平均精度得到了适度但有价值的提升，而过滤的结果则比较模棱两可。

如本节之前介绍的，Pitkow 等人 [1275] 取得了更大的收益，采用的是相当不同的评价方法。他们使用了相似的本体向量轮廓，但是他们没有给出确切的细节。他们的方法使用 677
了查询扩充和重排序。

2. 相关反馈方法

Teevan 等人 [1570] 描述了一些基于各种不同的用户轮廓的方法，为搜索引擎返回的前 50 个结果进行重排序。一个简单且收效甚大的方法是提升来自用户最近访问域的 URL 排序。一个更复杂的模型采用用户桌面搜索工具的索引来表示它们，即个人计算机上的所有文件、网页和电子邮件。桌面搜索的索引用作一个伪相关反馈引擎，它生成一个扩展的查询，用来对 Web 搜索引擎的前 50 个结果进行重排序。不幸的是，原始的网页排序和 URL 重排序都好于这个高度个性化的重排序。然而，一个结合了原始网页排序和个性化排序的混合方法可以改进原始的 Web 排序，这个改进比较小，但却是统计显著的。

Waern [1662] 研究的用户轮廓中包含索引项的长列表，它们或者是用户手动构建的，或者是自动产生的。研究发现，用户普遍无法改善机器学习的轮廓，但同时指出，用户在轮廓维护中的参与对于改正自动轮廓产生器的错误是十分重要的。

3. 通过用户的点击来刻画用户

我们已经讨论过 Joachims [841] 使用点击学习更好的排序函数的工作。Joachims 提到将点击用于个性化的可能性，但是没有报告结果。Dou 等人 [508] 应用 Microsoft 搜索引擎的日志对 5 个个性化搜索策略进行了一个大规模的评价，其中的两个是基于点击的，而其他的是基于自动产生的轮廓。他们发现个性化对于显著改善搜索质量是有潜力的，但这种改善在不同的查询间有很大不同，有时甚至有害处。他们发现那些有高点击熵的查询是从个性化收益最多的查询，简单的、基于点击的个性化始终是有益的，而试图捕捉用户兴趣的轮廓则不是很稳定。完全基于用户过去点击的方法只能改善用户之前提交的查询。为了解决这个限制，另一个方法使用点击模式将用户分配到有共同兴趣的组中，每个用户组的点击历史用于个性化。然而，所得到的个人和用户组的点击轮廓没有什么区别性。

4. 语言模型

Tan 等人 [1556] 描述了信息检索的语言模型框架的扩展，包括从点击行为产生的短期（当前会话）和长期的历史语言模型。这些历史模型可以看做另一种形式的轮廓。 678

5. 偏置的 PageRank

另一种完全不同形式的用户轮廓是由 Jeh 和 Widom [831] 提出的个性化 PageRank 向

量模型，在 15.6.1 节已经讨论过了。

15.6.6 隐式评价

Kelly 和 Teevan 在［896］中综述了来自用户在检索、过滤和推荐行为中的隐式评价。他们论文中的表 1 列出了 5 类用户行为——检查、保留、参考、注释和创建，它们可以在轮廓建立过程中观察和使用。表 1 也确定了每一类特定行为所作用的项目的最小范围，而他们论文中的表 2 将之前的相当多研究纳入到表 1 中。

Microsoft 的研究人员已经广泛地研究了隐式评价（从他们浏览器的测试版本得到）在改善 Web 搜索结果中的应用。Fox 等人［580］确定，一些隐式评价的概率组合，如点击和页面停留时间，可以准确地预测用户做出的显式判断。Agichtein 等人［18］扩展了这个工作，加入了查询依赖的测度，并提出了一个对于噪声健壮的分布式模型。Agichtein 等人［17］表明，当与大规模机器学习模型结合时，隐式评价可以用来改善 Web 搜索的性能，无论是通过对原始结果集的重排序，还是通过整合到基本的排序函数中。他们的研究包含3000个查询和 1200 万次用户交互。这些研究都没有以个性化的目的使用隐式评价，但是它们在构建个人或组别的轮廓时显然是有潜力的。鉴于机构之间的不同以及交互数据的稀疏，还不清楚这项工作的经验可以在什么程度应用到企业搜索中。

White 等人［1687］指出尽管显式的相关反馈可能是有益的，但它强加给用户一个负担。他们分析了相关反馈的隐式版本（Implicit version of Relevance Feedback，IRF），其中用户的一些行为，如阅读、滚动和保存，可以用于推断其相关性。尽管 IRF 不一定有益，但是他们报告，用户倾向于它，尤其是新手。他们还发现，IRF 对复杂的搜索任务更有价值，相比开始和结束，它更可能应用在搜索活动的中间阶段。

15.6.7 信息过滤

个性化搜索结果可以看成将信息检索（IR）和信息过滤（Information Filtering，IF）中的工具结合在一起。Hanani 等人［696］为 IR 和 IF 提供了一个详细的概念框架，并对它们进行了对比。首先产生通用的搜索结果，然后过滤那些用户不可能感兴趣的结果。个性化的目标是，通过将立即需求的简短叙述（查询）和更广阔的、长期的轮廓相结合，能够得到更好的随机搜索结果。在分流和报警系统中，我们仍然看到 IR 和 IF 技术的结合，但在这种
679 情况下，没有即时查询。相反，为搜索服务注册长期的轮廓。新创建或发现的文档与轮廓进行匹配，如果足够匹配，那么文档可以通过电子邮件或 RSS 转发给用户。

几十年来，Lexis-Nexis 等机构已经提供了信息的选择性传播（Selective Dissemination of Information，SDI）服务，用户注册一个包含布尔查询的轮廓，得到与过滤器匹配的所有文档。在这个模型中，用户需要创建一个过滤查询，并且必须维护它，以保证他们不会错过重要的文档或者为他们不感兴趣的文档付款。最近，谷歌在 Web 搜索引擎中提供了一个类似报警设施的提醒服务。对于用户注册的长期查询，排在前列的文档中，新出现的那些将作为候选提醒，发送到用户。谷歌研究人员 Yang 和 Jeh［1740］讨论了这个提醒服务的问题，并描述和评估了从用户搜索历史中自动抽取提醒轮廓的方法。面临的挑战是在查询日志中确定用户的长期兴趣，使得用户对看到新文档会有兴趣。

信息过滤的另一种方法是自动地将个人与一个组相关联，并使用组轮廓来定制结果。这就是所谓的协同过滤（Collaborative Filtering，CF）系统，有时称为社会化推荐系统（Social Recommender Sytem）。

15.6.8 社会化推荐系统

现代的搜索引擎，无论是在 Web 上还是在企业中，都在它们的基础排序方法中执行一类通用的协同过滤。被很多作者链接的文档，或者被很多读者标注或点击的文档，往往获得更高的静态得分，并在结果排序中有较高的排序。这样，个人搜索用户可以从作者、浏览器和搜索用户群体智慧中受益。Resnick 和 Varian [1343] 描述了一些推荐系统，以及它们是如何工作的。为了在协同过滤框架内完成个性化，人们可以在整个群体中识别组，并将个体关联到一个适合的组中。

Heer 和 Chi [740] 研究了对 xerox.com 网站上的用户会话进行分类的方法，并进行了一项用户研究，其中要求用户执行一些实际的信息发现任务。通过组合浏览路径和页面停留时间等特征，能够达到很高的聚类精度。但是，对于新的访问者或者新的浏览会话，分类有多快，以及分类是否可以用于改善搜索的性能，还不是很清楚。

一个在线购物网站能够向用户展示他们可能感兴趣的东西，吸引他们的注意，从而有效地增加销售。"购买商品 X 的用户也购买了 Y。"这个问题可以使用信息检索的方法解决，将消费者选择的商品（或购买商品的累计列表）看成一个查询，并检索相关的商品。然而，正如 Linden 等人在 [1036] 提到的，亚马逊发现基于搜索的方法在用户大量购买的时候会失败。相反，他们采纳关联商品的方法，使用一个离线计算的商品-商品相关性的矩阵。如果两个商品经常被同一个消费者购买，那么它们被认为是密切关联的。

有兴趣的读者可以参考 Adomavicius 和 Tuzhilin [15] 对于基于内容的协同和混合过滤方法的全面综述。 680

15.7 趋势和研究问题

企业搜索所面临的挑战是为知识密集型机构的全部文档内容提供单个查询的搜索界面。理想的企业搜索工具会提供充足质量的结果来支持额外的功能，如商业智能分析、用于执法的轮廓构造、知识挖掘、报告生成和多文档摘要。注意，企业搜索工具是这些功能的天然平台，因为它将来自多个资源库的文档和数据汇集在一起，并将它们转换成兼容的、可访问的模式。在最近几年，商业搜索产品更密切地支持或集成了商业应用。

然而，如 15.1 节指出的，有大量的证据表明，企业内部的搜索还没有接近当今 Web 搜索的用户满意度。员工很少访问企业搜索工具，跨越企业信息资源，返回相对于查询最有用的结果。这似乎很奇怪，虽然有很大的生产力和竞争力，但利益似乎从高效的企业搜索流走了。重要原因首先是对企业搜索的特定问题缺乏研究，其次是缺乏合适的企业搜索测试集，再次是公司文档和信息需求的保密性，以及机构间的巨大差异。现在慢慢有趋势开发至少覆盖部分企业搜索空间的测试集。希望保密问题可以很快解决，研究步伐可以加快。

在此期间，企业搜索中的一些重要领域的研究继续进行。第一，分布式信息检索，特别是个人元搜索，它与企业内部的异质信息资源的联合问题相关。第二，"在防火墙后面"，能够找到与 Web 搜索类似的、有效的排序因素。第三，对搜索结果个性化、定制化以及多样性的支持。第四，语言功能，如同义词检测、实体提取、翻译、摘要、查询建议。

面对企业搜索引擎的挑战，需要系统内所有组件的贡献，包括最初的创建和发布过程。虽然一直有进展，但在工程、研究、标准的采用和商业实践中需要更多的努力。

15.8 文献讨论

信息检索的普遍问题在其他章节已经讨论了，如爬取、索引、排序、结果展示、摘要和

多媒体检索，这些在企业搜索领域内部都非常重要。读者可以参考相关章节。本章主要关注企业搜索引擎的独特问题。

第一，机构内部的很多工程挑战［9，274，675，719，1535］一定要解决，以便获得高质量的文本文档来进行索引：根据机构部署的一系列资源库和应用程序进行适应性调整，以
681 便提取文档；有效地扫描包含文本文件的共享文件系统；准确和高效地从如 PDF 和 Office 文档等二进制文件中提取文本；对安全系统进行有效的身份验证。在某些情况下，从资源库中提取文档是不可行的，因此该资源库的搜索能力必须与主要的企业搜索工具联合。

第二，在异质文档集中排序文档和展示结果的问题。链接和锚文本等排序证据的宝贵来源，可能存在于某些子文档集中，而在其他中没有。据我们了解，对十分异质的文档集的单一索引检索进行优化的工作目前还很少。即使在关于分布式信息检索的广泛研究中，除了 Thomas 的博士论文工作［1580，1583］以及 **Stuff I've Seen** 系统［518］外，与真正的异质资源库联合相关的工作也很少。目前还不是很清楚，从人工划分 TREC 随机检索文档集而产生的模拟资源上进行实验所得到的结论在企业联合背景中是否实用。第三，当展示结果时，通常情况，必须对最初排序进行过滤来删除一些特定用户无权看到的文档。

第四，由于文档和信息需求的保密性，以及机构间在信息量、资源库数量、文档类型数量、搜索特点的巨大不同，评价企业搜索变成了一个困难的问题。这使得企业搜索的研究很困难，调试“开发中”（in the factory）的企业搜索引擎很困难。Hansen 和 Järvelin［697］、Freund 等人［589，590］，以及 Hertzum 和 Pejtersen［756］已经研究了真实的企业搜索，而 Craswell 等人［439］、Bailey 等人［126］则描述了面向企业搜索和专家发现的测试集。读者可以参考 2005—2008 年 TREC 企业搜索的任务综述和参与报告。这些可以在 http://trec.nist.gov/proceedings/proceedings.html 得到。

第五，Grefenstette 在 ECIR'09 的主题演讲［675］概述了 Web 搜索和企业搜索的 11 个具体不同。

个性化和定制化这两个重要主题并不是特定于企业搜索的，但在这个领域有一些重要的潜力和特殊的特征。除了本节上面引用的文章之外，［15，896，1262］等综述文章，“信息交互情境”（Information Interaction in Context）研讨会[⊖]的论文集也是开始阅读情境化和个性化论文的一个好地方。

两个专门的企业主题有很高的经济重要性，值得特别一提。基于公司记录的法律发现搜索有可能获得很高的影响力。Roitblat［1378］和 Baron 等人［149］提供了有用的概述。专利检索是另一个面向法律的任务，它对于很多主要的公司都很重要。自 2002 年的第 3 届研讨会开始，专利检索已经在日本国家信息研究所（Japanese National Institute of Informatics）主办的 NTCIR 系列研讨会中得到研究，参见 http://research.nii.ac.jp/ntcir/publication1-en.html 的在线论文集。最近，在维也纳的信息检索研究室（Information Retrieval Facility，IRF，http://www.ir-facility.org/the_irf）已经建立，目标是促进和支持开放信息
682 检索，特别是专利检索。它提供数据集和大规模的计算基础设施来支持研究项目，并先后赞助了 CLEF-09 中的知识产权检索和 TREC-09 的化学文献检索任务。

企业信息架构的广泛主题是 2006 年在 Morville 和 Rosenfeld 的书籍《Information Architecture for the World Wide Web》［1157］中提出的。《Search and Information Access Report》是一份回顾
683 了当前可用的企业搜索选项的报告，由 CMS Watch[⊜]不时发布。其他综述则关注于企业部门。

⊖ http://irsg.bcs.org/iiix2008/。

⊜ http://www.cmswatch.com/Search/Report/。

图书馆系统

——Edie Rasmussen 著

16.1 图书馆的信息环境

Web 是个奇妙的信息环境，能够提供范围广泛的主题信息。但它也是混乱和非结构化的，所提供信息的准确性、可靠性、完整性或者时效性可能都存在问题。在谷歌[㊀]上搜索“信息检索”，在第一页上显示如下链接：一篇介绍信息检索的维基百科文章，两个指向 1979 年 van Rijsbergen 撰写的关于信息检索书籍正文的条目，一门信息检索课程的主页，三个与信息检索期刊相关的网页，一个主要的人工智能协会的页面，由一家咨询公司提供的资源网页，以及《现代信息检索》第一版的主页。使用雅虎搜索得到类似的结果。显然，Web 搜索只是信息搜寻过程的第一步，而信息搜寻需要与用户进行高层次的互动，即用户持续点击链接，对所提供信息的类型和质量，以及是否与他的信息需求相关进行判断，直到信息需求得到满足，或者在可用信息范围内尽可能满足。

Web 是个快速、方便地回答问题（例如地址和事实），或者提供与主题相关信息的好资源。其信息存储的高度冗余意味着，在回答信息需求时，通常有许多可用的资源。但是 Web 并不能满足所有的信息需求。Web 固有的出版自由是电子出版时代的显著标志，这意味着任何人都可以针对某个主题发布信息，而不考虑其准确性，因此信息可能带有偏见甚至是不正确的。并非所有的信息都是免费提供的，例如许多图书和期刊信息受到版权保护，在 Web 上是不开放的，另外一些信息是私有的，例如商业记录。并非所有的信息都是以数字格式存储的。Web 上的大多数信息，特别是图像或视频（参见第 14 章）等非文本资料，没有以任何方式编目或索引。因此，Web 搜索不能回答或者不能完全回答许多问题。 685

在当前面向 Web 的检索环境中，人们很容易忘记信息检索系统还在其他环境中发挥着重要作用。在很多情况下，典型的 Web 搜索将是不可接受的——在搜索与某个医疗条件对应的诊断和治疗方法时，甚至会发生致命的后果。在律师搜索法律案例、先例和专利权，企业搜索公司信息并以此为基础进行财务决策，研究人员收集背景资料以支持他们的研究项目，大学生准备一篇平衡的分析性学期论文等许多情况下，需要搜索比 Web 更正式、更结构化的信息源。图书馆提供的信息检索系统能够搜索文档集、书籍、期刊、数字化资料和数据库。这些资料已被系统地获取和组织，以满足图书馆用户的信息需求。而且，即使这些图书馆中的信息能在 Web 上公开，在企业世界中仍然有大量的信息存储在封闭库中以供搜索。

尽管图书馆有时给人的印象是需要通过卡片目录访问陈旧的、尘土飞扬的书库，但它们却是最早使用信息检索系统的机构之一。这种使用方式在早期主要通过两种形式：由商业供应商提供对远程电子数据库的访问，为读者提供参考服务；在图书馆创建馆藏资料的目录记录，并提供搜索服务。最近又增加了数字期刊[㊁]（电子期刊）、电子图书、关于当地机构或历史的数字化资料、数字化课程、Eprints 和机构资源库等馆藏资料，甚至还包括选定的网站。

人们创造了“混合图书馆”一词来形容这个传统印刷资源和现代数字资源的组合。一个

㊀ 2007 年 4 月的搜索结果。

㊁ 在本章中，我们更多使用“数字”来描述在线资源，而不是“电子”，因为后者是不够准确的早期术语。

典型的定义 [799] 如下：

> 混合图书馆是“新的”电子信息资源和“传统的”硬拷贝资源共存，并组合成综合信息服务，通过电子网关访问的图书馆。它既可以像传统图书馆一样提供现场服务，又可以通过互联网或者本地计算机网络提供远程访问。混合图书馆与典型的图书馆网站有两点不同：一个是印刷和电子信息资源永久和平等的包容性；第二个是用可扩展的方式、为特定用户组聚焦和解释包括特定主题和通用对象在内的整个服务。混合图书馆的哲学假设是，图书馆提供各种有组织的访问，而不限于本地馆
> 686 藏，后者只是传播方式的一部分。

图书馆所面临的挑战是为无数种类的本地和远程资源创建集成和无缝的访问。图书馆的读者习惯于按 Web 的“搜索框文化”[251] 操作，他们期望信息检索免费、简单而且方便，能够立即访问资源的全文。图书馆的馆藏信息有不同的来源、格式，且需要订阅和许可才能使用，于是像 Web 搜索引擎一样提供集成访问就成为一大挑战。有些挑战与访问暗网时遇到的问题类似，需要通过数据库访问信息，而不是表层的网页。

图书馆检索系统有些与众不同的特点。它们提供一系列数据库供用户访问：联机公共检索目录（Online Public Access Catalogue，OPAC）提供图书馆的核心印刷和数字馆藏信息服务；商业文摘和索引服务；电子期刊；电子图书库；（数字）特藏；Eprints；以及机构资源库。虽然理想的方案是将这些数据库整合成单一的检索系统，但在实践中它们是通过多个独立的系统进行搜索的。图书馆网站通常可以作为一系列搜索服务的网关或门户，尝试建立统一的外观和感觉。另一个增长中的趋势是，使用 OpenURL 技术帮助建立不同的搜索系统之间的超链接——从图书馆目录到电子期刊，从书目引用到期刊文章的全文。

本章的重点是实际使用中的图书馆信息检索系统。在 16.4 节，我们也将简要地讨论组织机构内部的新兴信息检索应用——企业搜索，它与图书馆环境中的信息检索有一些相似之处，也有一些不同。

16.2 联机公共检索目录

图书馆目录是图书馆馆藏资源的传统接入点。今天，大多数图书馆使用图书馆集成系统（Integrated Library Systems，ILS）来管理目录和馆藏。ILS 的核心组成部分是联机公共检索目录（OPAC）。

图书馆目录作为图书馆的馆藏清单，被设计为发现馆藏资源的工具。多年来这一功能先是由卡片目录提供，后来是由计算机制作的书籍、缩微胶卷和缩微胶片的目录提供。尽管最早的联机目录是由一些连接到自动化流通系统的模块组成，提供的目录记录很简单，功能也非常有限，但它们自 20 世纪 70 年代开始应用于图书馆。那时的流通系统是现在所谓的图书馆集成系统的第一个组成部分。到了 20 世纪 80 年代，真正的联机公共检索目录已经实现。OPAC 系统最初是在大型（通常是学术性的）图书馆系统内部开发使用，后来由商业供应商
687 开发并用于统包系统[⊖]中。OPAC 系统采用标准化的记录格式，一般是 MARC 记录，使用最少量的主题信息（标题、少量主题词和分类编号）；与商业信息检索系统不同，它们从一开始就面向最终用户（图书馆读者）。

Hildreth [764] 将联机目录的历史分成三代。第一代 OPAC 系统主要是寻找已知项（known-item）的工具，通常根据作者、标题和控制号搜索，并含有相当短的、非标准的书目

⊖ 统包系统（turnkey system），也称为交钥匙系统，由软件（经常也有硬件）组成，通常根据特定的图书馆类型和规模开发。在系统限制范围内，可能提供某些定制方案以适应特定的图书馆。

记录。作为处于起步阶段的典型技术，它们基本上是旧技术（卡片目录）的自动化。第二代OPAC系统先后增加主题词和关键字搜索功能，具有基本的布尔搜索能力，并能够按主题词浏览。第二代目录系统也可以选择显示格式（例如，长度），改进了可用性（例如，对于初学者和专家提供不同的对话，更丰富的错误信息等）。Hildreth 认为第二代系统的问题包括失败的搜索、导航的混乱、主题索引词汇表的问题，以及过大的、组织不佳的检索结果集。

根据 Hildreth 的描述，第三代系统需要增强的功能包括搜索策略协助、集成自由文本和受控词汇表、可以扩充的编目记录、跨数据库访问、自然语言输入、个性化显示和上下文敏感的错误校正。然而，多年以来，图书馆目录仍然处在 Hildreth 所说的“第二代平台”。OPAC 发展创新的障碍包括开发新系统的成本，以及可靠客户群的需求。对图书馆而言，选择和迁移到新系统是昂贵的过程。由于预算总是被挤压，图书馆在选择未经试验的新系统时一直很谨慎。他们已经学会了警惕“下一版发布”综合症，而系统开发者却需要一个稳定的客户群来为新系统开发提供经费。

第三代系统包含着生活在前 Web 环境中的 Hildreth 未曾设想的功能。Web 上可用的电子资源已经使本地和全球资源、编目信息和其他电子数据库之间的区别变得模糊。自动化系统供应商可以从图书馆过渡到混合的数字/打印环境的过程中获得利益。在竞争激烈的市场中，他们的生存依赖于能否帮助图书馆在这种混合竞技场中获得成功。因此，最近许多图书馆系统的开发重点是在新的开放式系统架构下部署 ILS 功能。这些新系统的标准功能包括改进的图形用户界面（Graphical User Interface，GUI）、对 Z39.50 和都柏林核心标准（Dublin Core，多媒体资料的元数据标准）的支持、电子表单、超文本链接，以及 Java 编程功能。在基本布尔搜索功能之外，系统还引入了结果的相关性排序功能。在引用和图书馆资源之间自动建立链接的功能已被纳入。有些产品支持 10.7 节、11.10.3 节和 15.3.8 节所讨论的联合搜索或元搜索。

但是，由于发展缓慢，未能和其他环境的发展趋势匹配，目前这一代 ILS 已经受到了批评。例如，Breeding [252] 指出，虽然 ILS 能够很好地处理传统图书馆资源，但它们在电子内容产品方面已经落后，结果就成了模块的大杂烩，严重缺少集成。OPAC 模块的核心搜索能力也受到了 688
批评，有些系统不能对结果按照相关性排序，而另外一些系统虽有排序功能，但无法提供对用户有用的排序列表（参见 Schneider 在 ALA Techsource（美国图书馆协会技术博客）发表的帖子 [1441] 对这一点的有趣讨论）。今天，典型的图书馆自动化环境，特别对中、大型的学术图书馆而言，需要 ILS 管理传统的内容和一整套附加产品，以支持多种类型的电子内容。

对照 Web 的发展速度，图书馆系统的发展步伐引起了越来越多的不满。最近一篇题为“Rethinking How We Provide Bibliographic Service for the University of California”的报告中指出：

> 在图书馆前沿，我们的书目系统没有跟上环境不断变化的步伐。格式、工具、服务和技术的持续发展，已经颠覆了我们对馆藏资料的安排、检索和表示。我们的用户期待使用简单和回馈直接的系统，并且把亚马逊、谷歌和 iTunes 作为对我们评判的标准。我们目前的系统在它们周围黯然失色 [1618，p.2]。

16.2.1 OPAC 和书目记录

图书馆使用标准化体系对馆藏资料（文本和其他媒体）进行编目和分类，这有利于合作和一体化。通常情况下，它们遵循这样的一套做法；使用英美编目规则（Anglo-American Cataloguing Rules）描述这些资料，采用美国国会图书馆（Library of Congress）或杜威十进制分类法（Dewey Decimal Classification）等组织模式来指定主题代码，利用主题词表（如国会图书馆主题词表）来指派一系列的主题描述符。基于这种标准化体系，图书馆联盟可以进行合作编目，以降低图书馆馆藏资料的单位编目成本，并通过共享数据库扩大访问，

以促进资料的共享。因此，图书馆编目依赖于书目公益机构集中和共享信息，例如联机计算机图书馆中心（Online Computer Library Center，OCLC）[⊖]。OCLC 是由世界各地 112 个国家和地区的 69 000 多家图书馆组成的合作会员制组织，所维护的 WorldCat 来自 1 万多家图书馆的联合馆藏目录，包括了超过 1.25 亿条书目记录和超过 13 亿条图书馆馆藏信息。2006 年，此目录向公众开放，网址是 worldcat. org。通过所提供的 FirstSearch 服务，OCLC 也成为一家数据库供应商，具有表 16-5 所示的特性。

支持许多不同图书馆的联机目录库之间进行合作的基础是机器可读目录（Machine Readable Cataloging Record，MARC）。MARC 是一种数据格式，它实现了 ANSI Z39.2 信息交换格式（Information Interchange Format）和 ISO 2709 信息交换格式（Format for Information Interchange）等国家和国际标准。它也有一些在世界范围内广泛使用的变体，例如 USMARC 和 UKMARC 等。MARC 记录样本如图 16-1 所示。

This example can be identified as a record for projected material by code g in Leader/06, and more specifically as a motion picture by code m in field 007/00. This record illustrates the use of several MARC data elements to describe an archival motion picture, including: the use of character positions 09-/22 in field 007, and multiple occurrences of fields 007, 300, and 541 for the several versions of the motion picture being described. Other noteworthy data elements include: the use of field 017 (Copyright or Legal Deposit Number); field 040, subfield $e (Description conventions); field 257 (Country of Producing Entity for Archival Films); and field 510 (Citation/References Note).

LDR		*****cgm##22*****#a#4500
001		<control number>
003		<control number identifier>
005		19920513133548.3
007		mr#bf##dnnartnnac198607
007		mr#bf##dnnbdtnnac198607
007		mr#bf##dnnaetnnac198607
008		870505 s1918#### xxu055 ####### #####ml ####d
017	##	$aLP12321$bU.S. Copyright Office
040	##	$a<organization code>$c<organization code>$eamim
245	00	$a=M'liss /$cPickford Film Corp. ; supervised and directed by Marshall A. Neilan ; photoplay by Frances Marion.
257	##	$aU.S.
260	##	$aUnited States :$bArtcraft Pictures Corporation,$c1918.
300	##	$a5 reels of 5 on 2 (1988 ft.) :$bsi., b&w ;$c16 mm.$3ref. print
300	##	$a5 reels of 5 on 2 (1988 ft.) :$bsi., b&w ;$c16 mm.$3dupe neg.
300	##	$a5 reels of 5 on 2 (1988 ft.) :$bsi., b&w ;$c16 mm.$3arch pos.
500	##	$aCopyright: Famous Players-Lasky Corp.; 18Apr18; LP12321.
500	##	$aOriginally released in 35 mm..
500	##	$aBased on a story by Bret Harte.
508	##	$aPhotographed by Walter Stradling ; art director, Wilfred Buckland.
510	4#	$aNew York times film reviews,$c5-6-18.
510	4#	$aVariety film reviews,$c5-10-18.
510	4#	$aMoving picture world,$cv. 36.l, p. 894, 897, 1043.
511	1#	$aMary Pickford (M'liss), Theodore Roberts (Bummer Smith), Thomas Meighan (Charles Gray), Charles Ogle (Yuba Bill), Tully Marshall (Judge Joshua McSnaggley), Monty Blue (Mexican Joe), Val Paul (Jim Peterson), Winnifred Greenwood (Clara Peterson).
520	##	$aA western comedy-melodrama set in the mining town, Red Gulch, Calif. about the untamed daughter (Mary Pickford) of the town drunk (Theodore Roberts) who falls in love with the new schoolteacher (Thomas Meighan) who is accused of murdering her father and the situations that occur during his murder trial.
541	##	$3ref print$dReceived: 8-20-80 from LC film lab; $cgift;$aPickford (Mary) Collection.
541	##	$3dupe neg$dReceived: 11-20-79 from LC film lab; $cgift;$aPickford (Mary) Collection.
541	##	$3arch pos$dReceived: ca. 1958 from USDA film lab; $cgift, copied from 35 mm nitrate on loan;$aPickford (Mary) Collection.
650	#0	$aFrontier and pioneer life$zWest (U.S.)$vDrama.
650	#0	$aTrials (Murder)$vDrama.
700	1#	$aNeilan, Marshall A.,$d1891-1958,$edirection.
700	1#	$aMarion, Frances,$d1888-1973,$ewriting.
700	1#	$aPickford, Mary,$d1893-$ecast.
700	1#	$aRoberts, Theodore,$d1861-1928,$ecast.
700	1#	$aMarshall, Tully,$d1864-1943,$ecast.
700	1#	$aMeighan, Thomas,$d1879-1936,$ecast.
710	2#	$aArtcraft Pictures Corporation.
710	2#	$aPickford Film Corp.
710	2#	$aFamous Players-Lasky Corporation.
710	2#	$aPickford (Mary) Collection (Library of Congress)$5DLC

图 16-1 MARC 记录样本，来自 Edie Rasmussen，并得到网络发展和 MARC 标准办公室的许可

⊖ http://www.oclc.org/。

MARC 记录有三个部分：一个固定长度（24 个字符）的记录头标；地址目次区显示记录内每个字段的 3 位数字标识符，和字段的字符长度（未显示）；以及数据字段和子字段。子字段用子字段标识符（例如“$a”）表示，具体内容取决于每个字段。例如，260 字段包含出版发行信息，可能含有地点、出版商和日期等子字段（此处显示的记录已按字段标识符进行了格式处理，每个字段一行，以提高可读性）。Web 出现后，增加了 856 字段，显示数字资源的位置访问信息，在记录内部提供了超链接。

尽管 MARC 记录提供了有关馆藏资料的详细书目信息，但是大部分信息更适合于已知项搜索，而不是主题搜索。能够被索引从而支持主题搜索的基本字段包括 245 字段，即题名说明字段，以及 650 字段，即主题附加条目（Subject Added Entry）——论题性词语（Topical Term）字段，其中包含主题词或索引项。050—08X 字段也有以分类编码形式表示的主题信息，但在没有解释的情况下，大多数用户无法直接使用。为了在 MARC 记录中增加更多可搜索的主题信息，图书馆可以与出版商缔约，以便用书籍的目录（505 字段）和摘要和注释（520 字段）来丰富 MARC 记录。这些举措大大提高了 MARC 记录可搜索的文本量。

MARC 记录是书目信息交换的标准，ILS 使用它作为输入和输出格式，但记录的内部存储可能会以另一种格式。用于图书馆目录远程搜索的 Z39.50 协议在搜索和检索时兼容 MARC 格式，并允许同时查询多个 OPAC。如需了解更多信息，可参阅 6.2 节；如需了解 MARCXML，即 MARC 的 XML 版本，请参阅 6.4.3 节。

16.2.2　来自 ILS 的信息检索

图书馆通过 ILS 提供的目录包含了以元数据或编目信息形式表示的增值信息，这些目录可以方便用户访问。然而，对于大多数 OPAC 编目记录，元数据是唯一的可搜索信息，因为无法提供文档全文。OPAC 搜索有两种主要类型：已知项搜索和主题搜索。对于已知项搜索，目标是利用已知的信息，如作者或标题，为特定项找到完整的信息（通常是位置）。MARC 记录能很好地支持这种类型的搜索，因为它包含了非常详细的书目信息。由于编目记录是结构化或半结构化的，因此通过对作者或标题字段进行专门搜索，可以提高检索的性能。但编目记录对主题搜索的支持则较差一些，因为标识文档主题的可搜索文本可能只是有限的标题和主题词。

图 16-2 所示的搜索屏幕显示了一个学术图书馆目录的典型界面，它提供了关键字相关搜索、布尔检索，以及搜索特定字段的能力（这是基本的搜索屏幕，另外还提供进行更高级搜索的屏幕）。

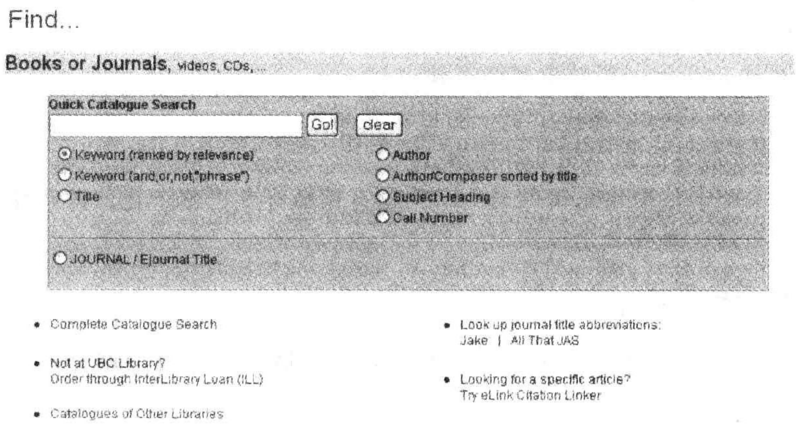


图 16-2　一个学术图书馆 OPAC 的界面，来自不列颠哥伦比亚大学图书馆网站

关键字搜索基于词的出现频率和所在字段的性质对结果排序。然而，由于只有有限的主

题文本以供搜索，相关性排序的效果可能没有全文数据库的结果令人满意。对于其他搜索选项（标题、作者等），按字段搜索不是在字段内搜索，而是需要从字段最左边的位置开始与查询项进行匹配，即需要精确匹配。例如，按标题搜索“信息检索”能找出以“信息检索”开头的所有标题，但却不会检索出“现代信息检索”。出于这个原因，结果列表可作为浏览的索引，并以查询项为访问点。对于大规模馆藏，这样的帮助可能不大。例如，输入“滑雪”为主题词进行搜索不会返回任何条目，因为“滑雪”并不是一个认可的主题词（虽然
689～691 “滑雪意外”是）——而认可的主题词是“滑雪板和滑雪”。甚至对浏览帮助也不大，因为在浏览索引时，“滑雪”和“滑雪板和滑雪”之间有许多页主题词。即使这种简单的查询表格，也需要对有关功能有很大程度的了解，这导致了一些长期存在的，用户与 OPAC 互动的问题。我们将在 16.2.4 节对此加以讨论。

最近一项创新是由北卡罗莱纳州立大学[1]实现的 Endeca[2] ProFind 导航引擎，该导航引擎常用于电子商务和其他企业的网站。它们从自己的 ILS 系统中导出 MARC 数据，利用搜索引擎建立索引，并允许相关排序和分面搜索（faceted search）。索引在夜间更新以加入新信息。从该图书馆的网站[3]上可以发现相关的技术信息。这些分面来自馆藏资料所标引的国会图书馆主题词，并允许利用主题、时段、地域、流派和格式等分面来改善原始搜索。在国会图书馆层次分类体系的基础上，该界面还提供了一个浏览选项。以上面的例子来说，输入关键字“滑雪”能返回元数据中包括“滑雪”字眼的文档列表，并按相关性排序，还可以提供一个分面列表来改善搜索。例如，87 篇文档使用完整的主题词“滑雪板和滑雪”，30 篇文档使用较窄的“越野滑雪”，用“娱乐性使用”细化搜索，则可以得到 13 篇相关文档。用户对该目录的初步反响是正面的。

692 显然，OPAC 检索功能的更大创新是必要的。加州大学的书目服务专责小组建议图书馆系统采纳以下的增强搜索和检索功能，以跟上当前数字环境的前进步伐：

- 提供用户直接访问条目的功能。
- 提供推荐功能。
- 支持定制/个性化。
- 为失败或不可靠的搜索提供替代选项。
- 为大的搜索结果集提供更好的导航。
- 在用户所在场所提供书目服务。
- 提供相关性排序，并利用全文。
- 对非罗马文字资料提供更好的搜索 [1618，p. 3～p. 4]。

针对 OPAC 重构，他们还给出了两个建议：

- 为所有的大学馆藏创建一个单一的目录界面。
- 支持对整个书目信息空间进行搜索。

这些建议涉及图书馆检索系统的另一个问题：馆藏存在巴尔干化（碎片化）现象，表现为离散的数据库集合，必须分别搜索。

16.2.3 混合图书馆的整合

图 16-2 所示的 OPAC 界面是典型的。它搜索已编目的图书馆馆藏资料，但对于图书馆

[1] http://www.lib.ncsu.edu/catalog/。
[2] http://endeca.com/。
[3] http://www.lib.ncsu.edu/endeca/technology.html。

用户可用的其他丰富资源仅提供有限访问。例如，界面提供了电子期刊标题搜索的选项。如果能够通过标题成功找到电子期刊，就可以通过链接离开本站，去访问期刊全文出版商或供应商的网站，如果图书馆有访问许可，就能让用户访问该期刊。由于电子期刊是打包绑定或者整合销售的，可能有指向多个网站的链接。如果许可证规定了用户访问电子期刊的权限（如特定机构的教师和学生），就需要一个验证步骤。在电子期刊网站上，出版商或整合者可能会提供搜索引擎，以检索该期刊或期刊组的文章。然而，除非只对特定的期刊或期刊集的资料感兴趣，这并不是主题搜索的有效方式，更可取的是搜索面向主题的数据库。

在这个学术图书馆的网站上可以访问的其他类型的资料包括电子书籍、政府出版物、数字数据、参考源，以及在线索引和数据库，每个类型都有自己的链接。有些资料是提供给所有人的，而有些资料来源则是有限制的，要求进行身份验证。正如提供界面的机构一样，由主流研究机构提供的丰富多样的数字和印刷信息同样令人印象深刻。图书馆的目标是：建立 693
跨越不同数据类型和数据集的联合搜索（federated search）机制。几乎所有类型的图书馆都在寻求联合搜索产品，这样的产品现在已经出现，有些作为主流 ILS 产品的一个组成部分，有些则通过与现有的图书馆系统同步的新一代接口提供服务 [255]。

该图书馆网站还提供了一个链接，指向对各种信息来源提供联合搜索的谷歌学术搜索（Google Scholar)。这些信息来源包括学位论文、Eprints、来自机构资源库的资料，甚至商业出版的电子期刊。但是，数据库是不全面的，因为它依赖于 Web 上的资料。这些资料有可能是不向公众开放的，虽然出版商可能允许谷歌的爬虫创建索引，但资料本身（除摘要以外）可能不允许未授权的用户浏览。谷歌学术搜索的结果按相关性排序，但可能是网站、期刊文章和书籍的混合体。个别图书馆可以将它们的 OpenURL 解析器链接到结果列表，允许授权用户从搜索结果中直接链接到相关的全文。

实现全面的主题搜索的途径是使用联机数据库。如 16.3.4 节所述，例子中的图书馆界面提供了指向许多联机数据库的链接。虽然有些是免费的，但大部分需要许可，并仅限于授权用户。其他数据库则可由馆员作为中介代表用户进行搜索。

为了方便联合搜索，由于引入了引用链接，搜索模块之间有了一定程度的集成。例如，如果在数据库检索中发现了用户感兴趣的文章，并且图书馆已经订阅了该文章所在杂志的电子形式，那么引用链接能够将用户从数据库所在的网站转到该杂志的出版商或供应商的网站，以访问该文章的全文。这是通过应用 OpenURL 标准实现的，这是一项 NISO 标准（Z39.88)，所提供的语法可以将元数据和对象标识符结合，并将用户指向解析器，以结合元数据和用户信息，将特定用户链接到适合的特定对象。

16.2.4 OPAC 和最终用户

OPAC 设计者面临的最大挑战也许就是创建可用的系统。每种类型的图书馆都有 OPAC，虽然研究型图书馆的用户可能对图书馆组织和访问信息的惯例有所了解，但是其他地方的最终用户可以是中小学生、大学生，或本地公共图书馆的读者，他们很少或根本没有接受过使用图书馆的正规训练，有些人只是偶尔使用 OPAC（Borgman 称之为“终生新手”[232])。然而，记录结构（MARC 记录）详细而复杂，组织结构（例如 LCSH 和国会图书馆分类体系）也远远称不上直观。

OPAC 搜索最常见的类型是主题搜索。用户进行主题搜索的失败记录有据可查 [159, 984]。常见的失败记录是空集（“零结果”），或者是另一个极端——信息超载，即检出的参 694
考资料太多，很难彻底检查。对南洋理工大学目录事务日志的一项研究 [984] 非常典型。研究表明超过 2/3 的搜索是关键字搜索。查询平均长度是 2.82 个查询项，只有 12%的查询

使用了布尔运算符。几乎 1/5 的查询没有返回任何结果。这些问题并不是新出现的。1986 年，Borgman 就提出了这样的问题："为什么很难使用联机目录?" 10 年后，她重新审视了这个问题："为什么仍然很难使用联机目录?"[232] 她认为原因是系统没有纳入有关用户行为的知识，要求搜索者提出查询表示的负担太重了。许多研究人员因此建议给搜索者提供更大的上下文帮助 [232，554，984]。

16.2.5 ILS:供应商和产品

ILS 市场是专门的，由占据市场竞争地位的、数量有限的供应商开发并销售产品。虽然很难找到没有 ILS 的图书馆，但图书馆仍然处在不断变化的状态，因为旧的系统过时了，或者不再受支持（遗留系统)，需要将系统升级或引入新的系统。大多数供应商的目标是细分市场：学术图书馆、公共图书馆、学校图书馆和专业图书馆。有关电子资源管理的新产品越来越受重视。表 16-1列出了三家供应商的资料。Breeding 评述了当前系统 [254]，并对 ILS 系统的市场和目前的发展进行了详细的讨论 [255]。

表 16-1 图书馆系统供应商

名称	描述
SirsiDynix Corporation URL：http://www.sirsidynix.com/	最大的 ILS 供应商，系统应用于 4 000 个图书馆。目前销售 Unicorn 和 Symphony 两个系统。最近的创新包括使用用户模式分析软件、联合搜索选项和 OpenURL 解析
Innovative Interfaces，Inc. (III) URL：http://www.iii.com/	目前提供 ILS 的第二大公司，它们的 Millennium 系统为学术、公众、专业和学校的图书馆提供服务。目前的功能包括相关性排序、登录认证、增强型门户功能，以及远程教育课程和机构资源库管理模块
Ex Libris 695 URL：http://www.exlibrisgroup.com/	一个面向学术图书馆及联盟的大公司，销售 ALEPH 和 Voyager 系统，Ex Libris 也提供 MetaLib 联合搜索系统和 Primo 高级接口。首创了跨数据库的链接引用机制

尽管早期 OPAC 系统是内部开发的，有时甚至是由热情的业余人员花费相当多的时间和金钱、冒着失败的重大风险开发的，但今天的图书馆环境支持第三方开发的统包系统，由少数企业和产品主导整个市场，尤其是较大的图书馆。MELVYL、Okapi 和 Cheshire 等系统主要是以研究为主，在学术图书馆使用，因此在初始经费使用完之后不能经常性地进行后续维护。然而，有些图书馆对于 ILS 开源软件的兴趣正在增加，虽然对于开发成本、支持、可靠性和功能仍有疑虑，但他们已开始尝试现有的开源系统 [253]。Koha 和 Evergreen（如表 16-2 所示）是两个交付使用的开源图书馆系统，有着不断增长的用户群。Evergreen 最初为佐治亚州公共图书馆开发，各种实现版本（Pines、Sitka）正在由加拿大和美国的图书馆联盟开发实现。Koha 软件起初由新西兰图书馆联盟开发，正在由国际程序员进一步开发。虽然这些软件包是开源的，但因为实现和进一步开发的需要，也提供商业支持。

表 16-2 图书馆开源软件

名称	描述
Evergreen URL：http://evergreen-ils.org/	Evergreen 是开源的、联盟质量的图书馆自动化软件，起初由佐治亚州公共图书馆为 PINES 网络开发，现已用于美国和加拿大各地的数百家图书馆。Evergreen 包括流通、编目、OPAC 以及统计报告等模块，采购和期刊模块也在计划中。Evergreen 的开发者已经成立了 Equinox 软件公司，提供支持和开发。Evergreen 的其他实现包括 Pines 和 Sitka（来自 Evergreen 网站）
Koha URL：http://www.koha.org/	最初是在新西兰开发，是第一个开放源码的图书馆集成系统。Koha 包括流通、编目、采购、期刊、预订、读者管理和分支机构联系等许多模块。Koha 在世界各地各种规模的图书馆中使用，是一个真正的企业级 ILS，具有全面的功能，包括基本选项或高级选项。它符合主流图书馆标准（来自 Koha 网站）

随着图书馆读者逐渐成为经验丰富的 Web 用户，他们对搜索和图书馆功能的期望值也在增加。作为社交软件和网络站点的用户，他们习惯于利用 CiteULike㊀、BibSonomy㊁和 696
LibraryThing㊂等网站来给他们自己的资源加标签和编目。

谷歌图书搜索（Google Book Search）㊃带来了对深度搜索的期望，即搜索书籍全文，而不是简单的书目描述。为了与这些热门网站竞争，图书馆推出新的界面，提供了 Web 2.0 功能，例如 RSS 订阅、用户标注和评论选项、联合搜索、导航帮助、相关排序结果和更强的视觉吸引力 [255]。虽然这些改进界面可能是现有 ILS 系统的一部分，但它们日益成为独立产品，并可作为使用图书馆 ILS 的前端。

AquaBrowser㊄就是这样一个例子，它提供的功能包括联合搜索、RSS 订阅，以及“我的发现”（My Discoveries）模块，允许用户给图书馆的资料加标签和评论。图书馆可以用目录和评论等附加内容来丰富书目描述。导航工具包括分面搜索和关键字云（word cloud）等可视化工具。已有的 ILS 公司也正在开发具有高级功能的界面，例如 Ex Libris 的 Primo、Innovative Interfaces 的 Encore。

16.3 信息检索系统与文档数据库

图书馆为各种各样外部产品的摘要和索引服务提供访问。这些产品一开始是印刷形式，图书馆替感兴趣的用户订阅。它们的电子形式通常称为数据库㊅：如果包含文章的引用（通常为摘要形式）则称为书目数据库；如果包含文章本身则称为文档或全文数据库。由于存在电子数据库，图书馆可以有这样的选择：在远程站点搜索由生产商或其他供货商提供的数据库，或者得到许可后把它们安装在本地。今天的图书馆考虑成本、用途和格式，提供这些选择的组合。

由于书目数据库含有大量的文本信息，因此其信息检索系统的初步开发受到了政府实验室的科研项目支持，目标用户是训练有素的搜索中介。计算机书目信息检索在 20 世纪 50 年代进行了首次展示，1964 年美国国立医学图书馆（National Library of Medicine，NLM）开始使用批处理。同样，在 20 世纪 60 年代，联邦政府资助了在线系统的原型开发项目，并在政府研究实验室得以实现。第一个这样的产品是洛克希德（Lockheed）公司 697
的 DIALOG 系统，它首先为美国航空航天局（National Aeronautics and Space Administration，NASA）实现，随后为其他政府部门服务，并在 20 世纪 70 年代初进行了商业化，所有权也发生了多次改变 [243]。今天的 DIALOG 系统通过互联网提供全球服务，数据库来自图书馆、其他组织机构以及个人。

除少数外，数据库供应商并不直接生产信息，而是通过共同搜索界面为用户提供搜索服务。数据库供应商从信息生产商那里获得数据库使用许可，加工数据库，引入尽可能多的标准化手段（如标准的字段名），建立倒排索引，安装数据库，创建标准格式的数据库描述和搜索者帮助，并为客户提供培训课程（见图 16-4）。他们通过设立多数据库的通用入口，提

㊀ http://www.citeulike.org/。
㊁ http://www.bibsonomy.org。
㊂ http://www.librarything.com/。
㊃ http://books.google.com/。
㊄ http://www.aquabrowser.com。
㊅ 文档数据库的开发商和供应商通常把他们的产品称为“数据库”，虽然它们缺乏关系数据库的表结构，而是由书目信息和文档全文组成。本章采取了这种提法。

供增值服务。数据库供应商还可以提供跨数据库的搜索。例如，DIALOG 允许搜索者同时搜索预定的数据库或者自己选择的一组数据库，创建一个合并的参考集，然后再删除重复记录。

16.3.1 书目和全文数据库

商业在线检索系统的历史起源于电子书目信息数据库的创建。印刷形式的文摘和索引工具在 19 世纪出现，在二十世纪则越来越普遍。专业团体、商业企业和政府机构作为出版商，从全世界的文献中选择相关资料，创建书目记录，并提供摘要和索引信息。这些数据库主要集中在科学领域，如《Chemical Abstracts》（化学文摘），《Biological Abstracts》（生物学文摘）和《Engineering Index》（工程索引），但人文学科和社会科学的产品也很快出现了，例如《Historical Abstracts》（历史文摘）和《PsycINFO》（心理学文摘）数据库。

随着文摘和索引的出版商转向电脑辅助排版并印刷它们的产品，信息磁带开始用于信息检索目的。今天，几乎所有的印刷文摘和索引产品都有电子形式，许多产品甚至只有电子形式，而没有印刷版本。由于存储成本大幅下降，许多电子数据库已经扩大到包括不仅文档的书目信息，还包含文档本身的文本。这样的数据库称为全文数据库，它们包括期刊文章和报纸数据库，以及作为参考资料的数据库，例如百科全书和目录。表 16-3 给出了一些 DIALOG 提供的常用数据库的特征。

表 16-3 DIALOG 收录的多个著名数据库的特征

名称	覆盖范围	规模
CA SEARCH: 《Chemical Abstracts》（化学文摘）	全世界化学及应用文献的书目记录，包括专利	2000 多万条记录，每周更新 18 000 条
MEDLINE	生命科学（特别是生物医药领域）书目记录，包括临床和实验医学、牙医学、护理学、药理学、精神病学和兽医学等。它对全世界 4300 种期刊建立索引	约 1500 万条记录，每年增加 40 万条
NY Times	《New York》（纽约时报）全文，包括 1980 年至今的杂志、书评和星期日专栏	280 多万条记录，每天更新
《PsycINFO》心理学文摘数据库	心理学和相关行为科学与社会科学国际文献的书目记录库，包括精神病学、社会学、人类学、教育学、药理学和语言学，覆盖 1700 多种期刊	260 多万条记录，每周更新 1500 条

16.3.2 数据库记录的内容

在一般情况下，书目数据库生产成本昂贵，因为它们需要经过严格的挑选，分析所覆盖的文档。有些数据库涵盖一组特定期刊的资料，另外一些则试图在规定的主题范围内全面搜集全世界各种格式的文献。每个条目必须首先检查是否与数据库目标相关，然后索引、摘录，并存入系统中。698

每个书目数据库是一个独特的产品，专门为满足特定研究领域及其用户组的信息需求而设计。因此，不存在一个统一的数据库记录内容标准。通常情况下，它包含记录键和书目数据等标记信息，如作者、标题、文档来源、摘要、索引词或分类代码等主题标识符。对于全文数据库，也包括文档的文本。图 16-3 和图 16-4 分别显示《BIOSIS PREVIEWS》（生物学文摘）和《Historical Abstracts》的数据库记录样本。请注意，主题描述使用的词汇表（描述符和代码）非常依赖于研究领域（在这种情况下是生物学和历史）。

2/9/1 DIALOG(R)File 5:Biosis Previews(R) (c) 2006 The Thomson Corporation. All rts. reserv.
0015888673 **Biosis No.:** 200600234068
Maximum body size among insular Komodo dragon populations covaries with large prey density

Author: Jessop Tim S (Reprint); Madsen Thomas; Sumner Joanna; Rudiharto Heru; Phillips John A; Ciofi Claudio
Author Address: Zool Soc San Diego, Ctr Reprod Endangered Species, San Diego, CA 92112 USA **USA
Author E-mail Address: timj@uow.edu.au
Journal: Oikos 112 (2): p 422-429 FEB 2006 **2006**
ISSN: 0030-1299
Document Type: Article
Record Type: Abstract
Language: English

Abstract: This study documents variation in maximum body size of Komodo dragons (Varanus komodoensis) among the four extant island populations in Komodo National Park and compares an indirect measure of deer density, the major prey item for large dragons, to differences in maximum body size among islands. The largest 15% of dragons from the large islands of Komodo and Rinca were significantly longer and heavier than the largest 15% of dragons on the small islands of Gili Motang and Nusa Kode. There was a 33% difference in snout vent length (SVL) between dragons found on Komodo and those found on Gili Motang, with mass varying by more than four-fold. Density of deer pellet groups between islands ranged from 5.86 +/- 0.75 groups per transect on Gili Motang to 20.73 +/- 1.02 groups per transect on Komodo Island. Maximal dragon SVL and mass was highly positively correlated with this index of deer density. Low prey density on the two small islands could constrain body size via energetic constraints. At present we can not deduce if insular body size variation has arisen through genotypic or phenotypic mechanisms.

Descriptors:
Major Concepts: Terrestrial Ecology--Ecology, Environmental Sciences; Biogeography-- Population Studies
Biosystematic Names: Cervidae--Artiodactyla, Mammalia, Vertebrata, Chordata, Animalia; Sauria-- Reptilia, Vertebrata, Chordata, Animalia
Organisms: deer (Cervidae)--prey; Varanus komodoensis (**Komodo dragon**) (Sauria)
Common Taxonomic Terms: Artiodactyls; Mammals; Nonhuman Mammals; Animals; Chordates; Nonhuman Vertebrates; Reptiles; Vertebrates

Geographical Name: Komodo National Park (Indonesia, Asia) (Oriental region); Gili Motang (Indonesia, Asia) (Oriental region); Nusa Kode (Indonesia, Asia) (Oriental region); Rinca (Indonesia, Asia) (Oriental region)
Miscellaneous Terms: genotype; phenotype; extinction; prey density; body size variation; maximum body size; snout vent length
Concept Codes:
07502 Ecology: environmental biology - General and methods
07508 Ecology: environmental biology - Animal
62800 Animal distribution
Biosystematic Codes:
85725 Cervidae
85408 Sauria

Biosis Previews(R) (Dialog® File 5): (c) 2006 The Thomson Corporation. All rights reserved.

图 16-3 《BIOSIS PREVIEWS》生物学文摘数据库的记录样本

9/9/175 DIALOG(R)File 39:Historical Abstracts (c) 2005 ABC-CLIO. All rts. reserv.
1683680 54-6856
SETTLING THE CANADIAN COLONIES: A COMPARISON OF TWO NINETEENTH-CENTURY LAND COMPANIES.
Browde, Anatole
Business History Review 2002 76(2): 299-335.
Document Type: ARTICLE
Abstract: Compares the performance of two British land companies-the **Canada** Company and the British American Land Company-chartered to sell land and encourage emigration to the colonies of Upper and Lower **Canada** during the 1820's-40's. The **Canada** Company was not only fiscally responsible but also fully aware of Canadian conditions. In addition, it engaged in strategic planning for its operations. The British American Land Company, on the other hand, was badly managed. It undertook little in the way of strategic planning, was managed solely for the benefit of the proprietors, and poorly understood Canadian conditions. The **Canada** Company accomplished its mission of facilitating emigration to **Canada** after 1815, while the British American Land Company failed in this endeavor. Based on record books for the Bank of England, **Canada** Company Papers in the Ontario Archives (Toronto), Colonial Office records in the Public Record Office (Kew), British American Land Company and **Canada** Company papers in the National Archives of **Canada** (Ottowa), and other primary and secondary sources; 89 notes. (H. M. Friedman)
Descriptors: Great Britain| **Canada|Emigration|Land (sale of)|Canada Company|British American Land Company|1820's-1840's**
Historical Period: 1820D 1830D 1840D 1800H
Historical Period (Starting): 1820's
Historical Period (Ending): 1840's

Historical Abstracts (Dialog® File 39): (c) 2005 ABC-CLIO. All rights reserved.

图 16-4 《Historical Abstracts》数据库记录样本，来自 DIALOG

如这些数据库记录所示，它们包含的主题信息有两种类型：所谓的“自然语言”或“自由文本”信息，包括在标题或摘要字段中，由索引人员标引的索引项或受控词汇表项。大多数数据库在描述符字段中包含索引项，这些索引项通常来自数据库相关的索引词典（一种正

式的结构化索引词汇表)。一个例子是专门为心理学领域主要数据库 PsycINFO 开发的
699 《Psychological Index Terms》(心理索引词汇词典)。其他类型代码或索引的使用与特定的数据库有关(例如,《BIOSIS PREVIEWS》的生物系统学代码,《Historical Abstracts》(历史文摘)的历史时段代码)。主题项标引是数据库的主要生产成本之一。虽然目前使用中的系统只是进行"机器辅助",而不是完全的自动索引,但是能够使用恰当受控词汇的自动索引系统对数据库生产商是很有价值的。

有关"自由文本"和受控词汇对检索性能相对价值的研究课题很早就已经开始,并且持续到现在。20 世纪 60 年代的 Cranfield 研究就已涉及这个课题 [1508],直到现在仍有研究人员持续加以研究。Lancaster [972] 和 Rowley [1390] 对这项研究进行了很好的综述。
700 明确的答案还没有找到,但后来的研究似乎表明两类索引的互补可以提升检索性能。

16.3.3 联机产业:数据库供应商

如表 16-4 所示,文档数据库的生产商和供应商之间存在着一种协作关系。在一般情况下,数据库生产商制作产品并授权给供应商。供应商(也可以称为聚合者或搜索服务提供者),向客户提供搜索软件和访问,以便他们能够从单一来源搜索多个数据库。

表 16-4 数据库生产商和供应商的角色

数据库生产商	数据库供应商
设计数据库结构	开发搜索软件
收集范围内的文献	从生产者获得数据库许可
输入标准形式的书目记录	对记录结构进行标准化(如有可能)
制作摘要(或者编辑作者的摘要)	装载数据库,创建倒排索引
用受控词汇表标引	更新数据库(每天、每周,或者每月)
按固定间隔更新文件	为搜索者提供文档
销售备份文件和更新给供应商	销售给客户
	提供服务和客户培训

经常有人提到联机数据库产业,因为数据库的生产通常是由企业、专业学会或政府在以营利或者收回成本的基础上进行。这些数据库生产商创建数据库产品,通常由第三方或数据库供应商出售或租赁给图书馆。数据库供应商的角色是从生产者获得数据库的使用许可,并给用户提供增值服务。数据库供应商提供了某种程度的标准化记录格式,创建索引(以倒排索引形式),并为搜索多个数据库提供了共同界面。著名的数据库厂商有 DIALOG、Lexis/Nexis、OCLC 和 H. W. Wilson 等,表 16-5 介绍了他们的基本情况。有些数据库生产商自己也提供搜索服务,造成了数据库产业某种程度上的纵向一体化,例如美国国立医学图书馆,通过 Web 免费提供其 Medline 数据库,H. W. Wilson 公司则销售自有的系列数据库。

表 16-5 一些数据库供应商

名 称	描 述
The DIALOG Corporation URL: http://www.dialog.com	DIALOG 的自我描述是"提供关键信息,推动科学、工程、商业和知识产权研究的全球领袖。"DIALOG 的 600 多个数据库和 15 亿独特记录是 Deep Web 的一部分。它提供了文档全文的 OpenURL 链接
Lexis Nexis URL: http://www.lexisnexis.com	LEXIS-NEXIS 销售法律和商业全文数据库。它们提供了超过 50 亿的可搜索文档,覆盖四万多个法律、新闻和商业来源,并声称访问可靠率高达 99.99%。LEXIS 提供法律研究产品,包括州和联邦的判例法、法律和法规的访问,而 NEXIS 涵盖新闻和商业来源

（续）

名　称	描　述
OCLC URL：http://www.oclc.org/firstsearch/	联机计算机图书馆中心（Online Computer Library Center，OCLC）起先是一个图书馆资料合作编目的书目服务，现在可以提供超过 80 多个数据库和成千上万在线期刊全文图像的访问。其特点包括指向馆藏文档的链接，以及馆际互借模块
H. W. Wilson Company URL：http://www.hwwilson.com/	H. W. Wilson 于 1898 年开始制作印刷的索引，现在为公共图书馆、学校和高校图书馆市场提供了 70 多个数据库。该公司不仅制作数据库产品，而且提供对它们的访问。通过图书馆网页上的一个链接，可以使用 WilsonWeb 系统搜索其数据库

这些主要的商业服务有一个很重要的方面。它们的数据库规模非常大，需要向许多用户同时提供快速可靠的服务。Lexis/Nexis 这样描述它们的计算复杂度 [1017]：数据库包括近 50 亿个文档，有 500 万订户，每年有超过 10 亿的搜索。它们在 6～10 秒钟内返回答案集，并声称平均可用性和可靠性大于 99.99%。

除了少数外，例如政府机构所制作的数据库，大多数数据库都是在盈利的基础上生产并提供访问的。原有的定价模式是根据使用情况、连接到特定的数据库的时间和打印的书目记录而收费。这就带来了对专业中介的需求，他们可以在最短的时间内进行有效的搜索。更高 701
的连接带宽导致收费依据从连接时间变成了处理时间。最终用户的搜索行为也发生了改变，从图书馆网站访问数据库，以及期望提供图书馆资源链接，转向了基于数据库许可和订阅的模式。

16.3.4　来自文档数据库的信息检索

早期在线检索服务的功能主要基于布尔检索模型，然而，在信息检索领域的研究重点是通过非布尔模型提高检索性能，如向量空间模型、概率模型和语言模型。许多因素导致了这些服务选择布尔模型。当时的索引和检索研究，尤其是 Cranfield 研究，进行了一系列自然语言和控制词汇的比较实验，发现“自然语言”检索的性能可以与人工索引检索性能水平相媲美。有些图书馆已经使用边缘切口卡片和光重合比孔索引卡等工具在人工检索中实现布尔逻辑，这似乎提供了基于文档中词的组合实施检索的自然机制。当时，其他检索模型的研究尚处于起步阶段，尚未在大型数据库上证明其有效性。或许最重要的是，当时的计算机只有有限的处理和存储能力，在线环境虽然足以支持布尔检索所需要的倒排索引结构和逻辑运算，却无法为其他更加计算密集的检索模型提供实时检索性能。

尽管信息检索研究表明，其他模型可能提供更好的检索性能，但布尔检索一直是数据库 702～703
供应商提供的最常见的访问方式。作为一种替代的访问方式，近年来的一些系统已经增加了自然语言输入和结果排序功能。布尔检索占据主导地位的原因包括经济上的考虑（搜索软件和数据库结构进行重大改变的成本）、服务问题（客户群已经按照现有系统培训），以及替代产品在操作环境中尚缺乏证据支持其可行性 [1318]。虽然现在的商业系统提供了一些自然语言搜索能力，但其进展似乎不大，在这些系统中，最常用的仍然是布尔查询，特别对搜索专家而言 [306]。

在一般情况下，数据库供应商使用针对具体系统的专用搜索软件，因此搜索多个系统的专业人员必须了解每个系统不同的命令词汇。数据库供应商也提供 Web 界面，这样就无须学习命令语言。但经验丰富的搜索者经常会发现界面响应缓慢、缺乏所需要的功能，因此更喜欢在命令模式下运行（见图 16-5 DIALOG 的 Web 浏览器例子，以及图 16-6 所示的命令

驱动搜索)。通用命令语言的一个标准是 NISO Z39.58 或 ISO 8777，但 NISO 标准后来被取消，或许是因为其功能可以通过 Web 界面提供。

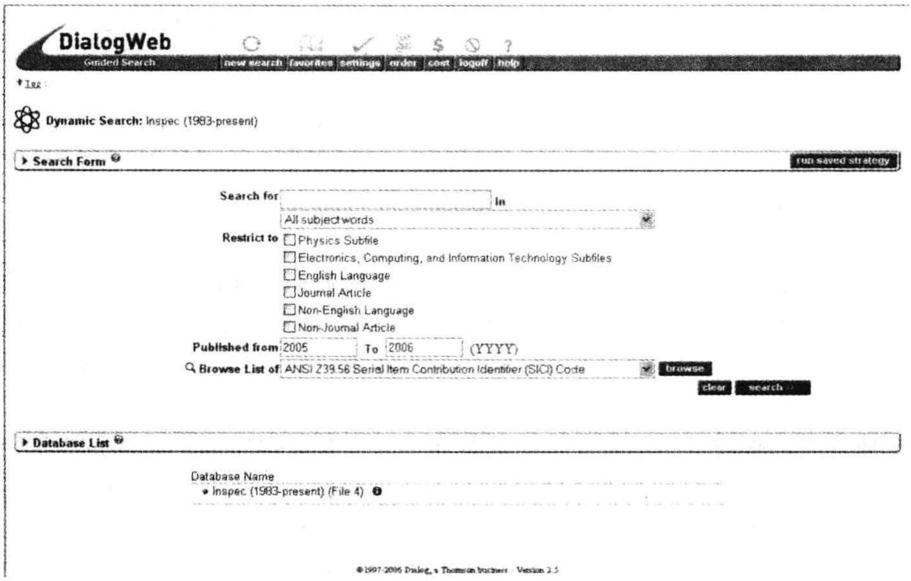


图 16-5 DIALOG/Web(基于表单的搜索屏幕)

书目信息检索系统的基本功能是搜索单一的项或词组，或者它们的布尔组合，以创建可进一步操作的文档集，然后打印或显示。通常情况下，系统也提供邻近操作选项，以定义查询项之间的关系(*A* 和 *B* 相邻、*A* 在 *B* 左右 *N* 个词范围内等)，并指定查询项在记录中的位置(*A* 出现在标题字段、*B* 出现在描述符字段等)，在这种情况下，倒排索引中位置信息
704 的存储量将更大。其他可用的功能包括浏览数据库索引以选择查询项，或者根据数据库同义词典项间关系发现候选查询项。与特定类别数据库有关的、更复杂的其他功能也是可用的，例如在化学数据库中搜索化合物结构的功能。

在搜索者输入查询项后，系统将创建由包含这个项的所有文档组成的相应“集合”，并分配集合编号给搜索者使用。在临时存储中，保留多个检索文档集。这些集合编号在发出搜索命令时作为文档集的代表。布尔逻辑可以用来处理现有集合。显示命令允许搜索者查阅搜索历史记录，并返回以前的结果集。根据查询项或表达式返回的结果集大小，并查阅相关文档及其索引，搜索者不断修改搜索，直到他们觉得已经获得最好的结果。这种结果集的迭代改进过程是科学也是艺术，其成功高度依赖于搜索者的技能和学科知识。

图 16-6 显示了一个典型的 DIALOG 布尔搜索。在此搜索中，用户请求一个特定的数据库(文件 4，INSPEC：The Database for Physics，Electronics and Computing(物理、电子和计算机数据库))，然后使用“Select”或“S”命令来创建记录集。集合 1 包含标题、摘要或描述符字段的任意位置含有“WWW”或“Web”的所有文档。“(w)”或“()”表示邻近操作，所以集合 2(S2)将包括含有“information foraging”或“information scent”的所有记录。这两个集合进行布尔与(AND)操作，得到 23 条记录，再进一步限制到标题(ti)或描述符(de)字段包含查询项的记录，最终得到 18 条以供审阅。同一个搜索可同时在多个数据库中进行，并可用重复数据删除命令来消除重复记录。

```
BEGIN 4
File    4:INSPEC 1983-2006/Jun W4
        (c) 2006 Institution of Electrical Engineers

      Set   Items  Description
      ---   -----  -----------
?

S WWW OR WEB
             8819  WWW
            59907  WEB
      S1    63381  WWW OR WEB
?

S INFORMATION()(FORAGING OR SCENT)
           566959  INFORMATION
              435  FORAGING
               79  SCENT
      S2       35  INFORMATION()(FORAGING OR SCENT)
?

S S1 AND S2
            63381  S1
               35  S2
      S3       23  S1 AND S2
?

S S3/TI,DE
      S4       18  S3/TI,DE
?
```

图 16-6 一个 DIALOG 搜索（命令行界面）

为了确保市场地位，数据库供应商开发可能对客户群有价值的新功能，并添加新的数据库产品。在一般情况下，这些新功能将扩充到现有的布尔搜索引擎——重复记录的消除、检索集内的复杂排序等。然而，几个主要的数据库供应商在他们的系统中加入了“自然语言”搜索功能。例如，DIALOG 提供了 TARGET，LEXIS-NEXIS 有一个名为 FREESTYLE 的系统。FREESTYLE 接受自然语言查询，TARGET 要求搜索者删除对搜索无用的查询项。这两个系统都提供检索文档的排序列表。这些“自然语言”系统是标准布尔搜索的辅助模块，并不是要取代它。一个 TARGET 的搜索例子如图 16-7 所示。

这个搜索针对 BIOSIS 数据库。系统首先向搜索者提供了一系列指令，以便处理短语和同义词等。在“?”提示符之后，搜索者输入一系列查询项（最多 25 个），在这个例子中是“komodo dragon food diet nutrition”。默认情况下，搜索限制在近两年的文档内，并按排序显示 50 个得分最高的纪录。这个例子对查询项没有任何限制，但屏幕上的说明显示，可以对查询项进行布尔逻辑操作，产生一个带排序功能的布尔搜索结果。搜索者显示了排在最前列的三篇文档，请注意，这些文档都不包含“komodo”或“dragon”，但根据 TF-IDF 加权方法，含有这些项的文档应该排在更前列。这表明排序算法的局限性，另 705
外，数据库也仅限于近几年，这些或许可以解释为什么这种自然语言功能并不更受欢迎。

数据库生产商和供应商提供的服务成本很高，他们发现自己在与 Web 竞争，而很多用户认为 Web 虽然有些杂乱无章，但却是现成的免费信息。信息产业的供应商被迫将目标用户限定于有需要的、并愿意为经过系统组织的高品质信息支付费用的用户，他们通常来自研究和商业环境。

```
? b55
File  55:Biosis Previews(R) 1993-2007/Mar W2 (c) 2007 The Thomson Corporation

? target
Input search terms separated by spaces (e.g., DOG CAT FOOD). You can
enhance your TARGET search with the following options:
  - PHRASES are enclosed in single quotes (e.g., 'DOG FOOD')
  - SYNONYMS are enclosed in parentheses (e.g., (DOG CANINE))
  - SPELLING variations are indicated with a ? (e.g., DOG? to search DOG, DOGS)
  - Terms that MUST be present are flagged with an asterisk (e.g., DOG *FOOD)

Q = QUIT   H = HELP

? komodo dragon food diet nutrition

Your search will retrieve up to 50 of the statistically most relevant records.
Searching 2006-2007 records only ...Processing Complete
Your search retrieved 50 records.

? [viewing records; customized browse requested]

------- Item:  1 -----------------------------------

Indirect estimates of net acid excretion and net rate of endogenous
non-carbonic acid production in the Young British population: analysis
of the National *Diet and *Nutrition Survey aged 4-18 years.

- Statistical Relevance:  93\%

- Term Frequency: KOMODO - 0 ; DRAGON - 0 ; FOOD - 1 ; DIET - 3 ; NUTRITION - 3

------- Item:  2 -----------------------------------

Vegetarian *nutrition : Preventive potential and possible risks.
Part 2: Animal foods and recommendations

ORIGINAL LANGUAGE TITLE: Vegetarische Ernahrung: Praventives Potenzial und
mogliche Risiken
Teil 2: Lebensmittel tierischer Herkunft und Empfehlungen

- Statistical Relevance:  88\%

- Term Frequency: KOMODO - 0 ; DRAGON - 0 ; FOOD - 2 ; DIET - 6 ; NUTRITION - 3

------- Item:  3 -----------------------------------

A pair-feeding study reveals that a Y5 antagonist causes weight loss in *diet
induced obese mice by modulating *food  intake and energy expenditure

- Statistical Relevance:  87\%

- Term Frequency: KOMODO - 0 ; DRAGON - 0 ; FOOD - 3 ; DIET - 2 ; NUTRITION - 1
```

图 16-7 DIALOG 的一个 TARGET 搜索

16.4 组织机构内部的信息检索

与图书馆一样，公司和非盈利性组织需要处理许多不同媒体和不同格式的文档，很多信息是独特的和专有的。有些信息资料可能存储在关系数据库中，但大多数是非结构化的文本，这正是信息检索系统的处理对象。基于 Web 搜索的经验，人们期望通过一个单一的界

面快速高效地查找企业信息。然而，这些期望没有在组织机构内部得到满足，并有证据显示，员工花费大量的时间搜索，却往往无法找到开展工作所需的信息 [553]。这些要求导致了一个新市场，即在组织机构内部，通常是公司内部网使用的信息检索系统。 706~707

这种信息检索应用通常称为企业搜索。Hawking [719] 将企业搜索定义为：在组织机构内部搜索任何数字化的文字资料，包括搜索对外的网站、公司内网，以及所拥有的诸如电子邮件、数据库记录和共享文档等任何其他电子文本。他指出，检索环境的复杂性意味着市场上只有数量有限的检索产品有能力在组织机构内部运营。

企业搜索的特性使得它成为信息检索系统设计师所面临的一项挑战 [274，1162]。信息可能是结构化的，也可能是非结构化的。文档由多种来源产生，可能有许多不同的语言，没有通用的格式标准。元数据可能会根据一些不同的模式来创建，也可能根本没有。并非所有用户对所有信息都具有相同的访问权限，员工记录等一些信息是高度机密的。联合搜索意味着必须从各种来源和格式的数据中创建一个单一的排序列表。不同的情境可能需要不同的排序方法。

虽然已有一些从 Web 搜索转移到企业搜索的技术，但两种应用之间存在着多个重要的差异。在 Web 上有大量的冗余信息，用户往往只需要寻找令人满意的任意文档，而不是特定的某一篇。企业搜索的用户是有特定信息需求的员工，往往只有一篇特定的文档或文档集能满足他们。这些文档内的链接结构是有限的，不能以同样的方式使用 PageRank 或 HITS 等网页排序算法。用户不在乎文档是否受欢迎，只关心是否有他们做好工作所需要的信息。但从积极的方面看，组织机构内部产生的内容通常是可靠的，所以垃圾邮件和优化技术不再是问题。

企业搜索软件通过爬取企业内网发现和索引文档、电子邮件、数据库和其他信息资料。这一点类似于 Web 搜索软件。在意识到企业搜索所面临的挑战后，TREC 自 2005 年起开始举办企业搜索评测，起初以电子邮件作为数据类型，后来加入公共机构的网页 [439，126]。因为企业搜索是与特定任务相关的，所以在这种环境中用什么指标进行评价是一个棘手的问题。

第 15 章已经详细地介绍了企业搜索系统，附录 A 介绍并比较了许多开源搜索引擎。

16.5 趋势和研究问题

图书馆馆员和图书馆用户要求新的和改进的混合图书馆信息检索系统。不管信息检索系统是由数据库供应商还是统包 OPAC 系统提供，除少数外，馆员都是信息系统的主要消费者。即使在数字图书馆环境中，他们强调的也是提供多种信息检索模块的集成访问。因此，他们的兴趣是获取和使用在自动化环境中易于集成的系统，方便自己和读者使用。Yu 和 Young 表明 [1757]，图书馆的用户从 Web 经验出发，欣赏信息检索系统。与此同时，为满足这些期望，混合图书馆的发展已越来越具有挑战性。在 Web 时代之前，OPAC 708
是图书馆馆藏的关键入口，而由图书馆馆员作为中介，搜索由多个主要供应商所提供的数据库来进行补充。但如今这些已经让位于多种格式和来源复杂的信息环境。越来越多的人感觉到，图书馆所提供的信息系统不足以帮助他们完成任务。图书馆、书目服务和研究人员需要改进搜索功能、更可搜索的文本、集成的系统，以及能更好地满足用户搜索行为的系统 [159，307，764，1618]。

企业搜索的一个主要问题是如何将信息检索解决方案适用于复杂的企业环境。一个重要的研究课题是如何用与组织目标相一致的方式来衡量企业搜索引擎是否成功，如何开发组织

机构的分类词典也是一个众所周知的问题。针对组织机构内部文档，研发数据挖掘和信息提取技术有着显著的经济价值。

16.6 文献讨论

很早就有关于联机数据库和系统的有趣文献，Bourne 和 Hahn 的《A History of Online Information Services，1963-1976》[243] 是绝佳的开始，而 Neufeld 和 Cornog 则介绍了电子数据库的历史 [1197]。

Kochtanek 和 Matthews 的《Library Information Systems：From Library Automation to Distributed Information Access Solutions》对图书馆的信息系统进行了概述 [914]。《Library Technology Reports》[⊖] 现在是一个图书馆技术信息的好资源，每年出版 6 期，每期是一份讨论特定主题的报告，例如链接和 OpenURL、图书馆开源软件库，或者下一代图书馆目录等。

Bates 的综合报告 [159] 是一份文献综述，讨论与图书馆目录相关的信息搜索行为，以及访问词汇表在 OPAC 检索中的作用。加州大学的报告《Rethinking How We Provide Bibliographic Services for the University of California》[1618] 很好地研究了目前的文献和
709 实践，并给出了未来 OPAC 搜索和检索的路线图。

⊖ http://www.techsource.ala.org/ltr/。

数字图书馆

——Marcos Gonçalves 著

17.1 介绍

在信息检索的相关领域中，数字图书馆（Digital Library，DL）是最先进、最复杂的系统。除了标准的搜索和浏览功能外，数字图书馆通常提供许多其他的增值服务，如文档保存和推荐、参考咨询服务、选择性信息传播服务等。此外，所有这些服务都覆盖各种类型的多媒体数据（例如，音频［464］和视频［1084，1695］），并能以分布式方式提供。

数字图书馆在电子出版时代随着 Web 一起诞生，因此吸收了 Web 的许多精髓，例如，免费提供在线学术内容的开放获取运动（Open Access Movement，参见 17.5.2 节）就是在数字图书馆社区内出现的。然而，数字图书馆与 Web 在某些重要方面有所不同。数字图书馆中的信息有明确的组织、描述和管理。因此，输入通常受到更严格的控制，目的是提高质量。与缺乏焦点的一般 Web 相反的是，数字图书馆通常针对有特定信息需求和任务的特定用户社区。由于更加聚焦，因此数字图书馆可以提供更加专业化的、面向特定社区的服务。这意味着，为了增加成功的机会，这些社区应在各方面参与数字图书馆的活动，包括规范化、创建和使用等。最后，与 Web 的低归档性质相反的是，保存是（或者应该是）数字图书馆的一个关键方面。

传统图书馆的关注焦点是物理对象，相比之下，数字图书馆几乎完全消除了物理世界固有的访问和传播限制。此外，数字图书馆有彻底改变传统出版方式信息链的潜力。在此背景下，作家、评论家、编辑、出版商、图书馆员和档案管理员的角色和责任变得模糊，用户在 711
不同的时间承担不同的角色，数字图书馆成为所有参与者之间（直接）互动和沟通的主渠道。然而，所有这些优势并非没有风险和成本。最重要的是，没有明确的实体来负责保存所有正在创建的数字资料，以及相关的知识产权、版权管理、条款和条件等法律问题。

因此，虽然其成功主要取决于信息检索技术［1434］，但数字图书馆还包括许多其他技术、方法论和过程。本章介绍数字图书馆的基本概念，包括数字对象和馆藏、元数据和目录、资源库/档案库和服务，以及社会经济问题、软件系统、案例研究，还包括数字图书馆领域的研究挑战。

17.2 定义数字图书馆

定义数字图书馆面临的挑战之一在于，涉及这个领域的不同社区有不同的角度。Borgman［234］明确指出，无论在研究领域还是在实践领域，数字图书馆领域都存在不同甚至竞争的视角，因此对定义术语、刻画术语和建立语境等活动造成了巨大的困难。

研究人员从不同的角度，以不同程度的覆盖度和广度（例如［129，915，1340］），对定义数字图书馆做了许多尝试。我们在此说明两个例子。

Witten 和 Bainbridge［1704］的定义：

> 数字图书馆是由数字对象组织成的集中馆藏，包括文本、图像、视频、音频，访问和检索馆藏的方法，以及馆藏的选择、创建、组织、维护和共享。

这是一个以技术为主的定义，忽略了数字图书馆包含的社会方面。

另一方面，Waters [1668] 认为：

> 数字图书馆是提供资源的组织，包括专门的工作人员，对数字馆藏进行选择、构建，提供知识访问、解释、分发，保存其完整性，并确保其持续时间，供所界定的社区或社团以经济的方式使用。

这是明显不同的看法。

虽然远远称不上一致同意，但我们将给出自己的观点。我们认为：

> 数字图书馆是复杂的信息系统，能满足用户（或用户社团）的信息需求，提供信息服务（可以通过使用场景描述）、组织（通过结构）、展示（通过空间），并以可用的方式与用户沟通（通过流）信息。

我们的定义受到 5S 框架的流（Streams）、结构（Structures）、空间（Spaces）、场景
712 （Scenarios）和社团（Societies）等基本思想和概念的启发 [643]。流是任意项（如位、字符、像素和图像）的序列，代表数字图书馆的内容。结构可以视为带标签的有向图，对数字图书馆的内容进行组织。空间（例如，向量、概率）是服从一定运算约束的集合。空间用于服务支持和展示目的。情景可以看做是故事，它描述服务的行为，由能够修改计算状态的事件或行动序列组成，目标是完成一个功能需求。社团是实体和活动的集合，以及它们之间的关系。5S 框架首次全面形式地刻画了数字图书馆的基础 [643]。从形式化的基本概念出发，可以形式化定义和刻画最小数字图书馆⊖的基本概念（例如，数字对象、元数据规范、馆藏、目录、资源库和服务）。

注意，这个定义涵盖了数字图书馆的某些重要方面：1）应支持多种类型的媒体；2）信息在数字图书馆内明确组织；3）可以支持丰富的使用场景、信息服务和互动性；4）数字图书馆存在目标社区（或社团）。作为最小定义，不包括其他重要的方面，如保存和知识产权问题。5S 的开阔视野将贯穿本章的其余部分。

17.3 通用架构

数字图书馆存在多种架构，其中一些将在本章完整地讲述。在图 17-1 中，我们给出了数字图书馆的一个通用参考架构，仅供讨论目的。正如我们将看到的，其他架构可视为这一基本框架的变体。

数字图书馆由数字对象（例如，数字文档和图像等）的馆藏（collection）组成，并采用元数据记录（metadata record）的目录（catalog）来描述、组织或者指定馆藏对象可由何人使用。在理想情况下，每一个对象在目录中应当有一个对应的元数据记录，这个记录的具体结构由模式（schema）定义。这些对象通常是一起存储在具有馆藏和目录访问和管理能力的资源库（repository）中。服务（例如，创建数字对象或元数据记录、保存内容、提供增值服务，以及满足信息需求）建立在资源库上，并由社会环境中的使用者使用。服务还可以根据重用或者扩展能力进行合作，由简单的服务创建更高级的服务。数字图书馆通常提供简单的搜索和浏览服务，两者由索引服务加以支持，这就构成服务的最小集。用户界面作为“黏合剂”，来组织和显示提供的所有服务。

713 这里所描述的架构中的概念都是基本的，接下来将要详细讨论这些概念。

⊖ 定义数字图书馆的最小概念集合。

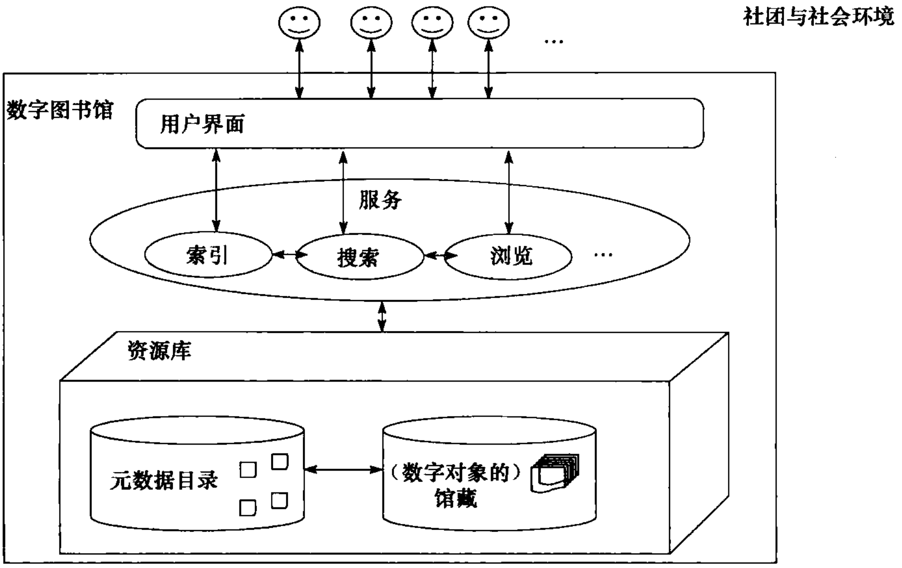


图 17-1 数字图书馆的通用架构

17.4 基本概念

17.4.1 数字对象和馆藏

数字对象可以视为数字图书馆的核心，因为它们承载了数字图书馆里的（大部分）信息。最简单的观点认为，一个数字对象是由内容和句柄两部分组成的，其中句柄是唯一的全局标识符 [427]。这样一个简单定义的扩展可能涉及伴随对象的、简单的描述性元数据（见 17.4.2 节）[958]、表示对象内部组成的结构化元数据 [643，1025]，以及产生对象不同视图或显示方式（例如，缩略图、同一幅图像对象或 Postscript 的不同分辨率版本、数字文档的 Microsoft Word 或 PDF 版本）[1249，962] 的复杂行为 [1193，1194]。下面，我们将讨论数字对象涉及的多个方面。

1. 句柄

一旦找到所需要的对象，就需要一种方式来引用它，以便从资源库中访问。引用或归档等其他许多情况下也有这种需要。句柄是为这个目标服务的数字对象唯一标识符。

在过去几年中，已经发展了许多语法和机制来提供数字对象的句柄或标识符。其中一些包括： 714

- 统一资源标识符（Uniform Resource Identifier，URI）：用于确定抽象或物理资源的紧凑字符串 [191，193，1113]。Web 术语“统一资源定位符”（Uniform Resource Locator，URL），是 URI 的一个子集，指通过其主要访问机制（例如，他们的网络“位置”），而不是资源的名称或者其他一些属性来标识资源。术语“统一资源名称”（Uniform Resource Name，URN），也是 URI 的子集，指的是需要保持全局唯一性的持久资源，即使该资源不再存在或变得不可用 [193]。
- 数字对象标识符（Digital Object Identifier，DOI）：一个基于非专用标准的开放系统，它提供了一个可互操作的机制，以便在数字环境下标识和交换知识产权 [1245]。DOI 遵从 URI 规范，并以经过检验的数字对象架构和知识产权管理标准为基础，提供一个管理知识内容的可扩展框架。
- OpenURL：传输与 URL 中一个或多个资源有关的信息（元数据和标识符）的标准语法 [1245]。

- 持久 URL（PURL）：指向中间解析服务、而不是互联网资源直接位置的标识符（或 URL）。这样的解析服务能够将标识符重定向到实际 URL，使客户能以正常方式完成 URL 事务。注意，虽然 PURL 允许不同的 URL 与它相关联，但 PURL 本身没有变化，使得它能够持久 [1305]。

2. **数字化**

许多数字对象并非天生就是数字化的，而是由真实对象或其代理（例如，现有人造物的图像）通过数字化进程创建的。这些数字对象，以及与派生它们的实际对象之间的明确关系，经常会被忽视。5S 扩展框架建议考虑这些关系 [1459]。大规模文档数字化的一个例子是世界数字图书馆（World Digital Library）项目。该项目受到谷歌和美国国会图书馆（Library of Congress，LoC）的支持，拟从 LoC 和其他国家图书馆的馆藏中，扫描善本、手稿、地图、海报、邮票和其他资料。截至 2005 年 6 月，谷歌已经数字化了大约 5000 部书籍，作为试点项目的一部分，目的是完善扫描技术，为脆弱的书籍建立副本而不破坏它。谷歌也有自己的数字化项目“谷歌图书（Google Books）”[656]，其合作伙伴是斯坦福大学、哈佛大学、密歇根大学、牛津大学等大学的图书馆，以及纽约公共图书馆。谷歌图书可以提供扫描图书的全文搜索，但由于版权问题，受保护的资料页中仅有少数能够供用户阅读。许
715 多书籍都使用书籍扫描机器人对书页进行数字照相，其速度高达每小时 1000 页 [897]。

3. **智能（数字）对象**

SODA（智能的对象，无言的档案）项目 [1193，1194] 主张将传统上与资源库或档案库相关的部分功能，例如执行访问限制或保留给定对象的使用历史，下推到称为桶（bucket）的智能对象。桶是集成的、智能的、面向对象的结构，包含数据、元数据，及它们的访问方法。例如，装载技术报告的桶，除了可以装载报告本身的内容之外，还可以装载用于生成结果的任何数据，以及产生它们的软件方法。其优点包括可移动性、自满足性和资源库独立性。

桶的功能包括：存储、跟踪和执行其使用条件；维护、显示，以不同的格式传播其内容；维护其事件日志。使用桶的目的包括：在单一的集成对象内，提供访问与数字对象相关联的各种数据（例如，关联数据、元数据和模拟等）的能力；以及自主性，即对象能够在数字图书馆环境之外存在且生存。

4. **馆藏**

从最简单的视角，可以把馆藏视为数字对象的集合。有人提出了支持收集和丰富馆藏的建设流程 [128，287] 的工具技术，允许最终用户来控制该流程 [1705]。其他人主张利用 Web 内容的广度优势，主要使用主题或聚焦爬虫，来建立数字图书馆特藏 [188，1242，1306]。Citeseer 系统 [386] 就是这样一个例子，它从 Web 上收集计算机科学的出版物，自动提取引用和元数据信息 [694]，甚至包括致谢内容 [431]。另一个例子是谷歌学术搜索 Scholar（Google）[657]。对于基于 Web 的馆藏生成，Web-DL 项目 [316] 提出了一个完整的架构和流程，包括信息收集、提取和发布，以及完成这些任务的具体工具。此外，也有相关工作试图使用聚焦爬虫寻找数字图书馆缺失的文档 [1790]。

17.4.2 元数据和目录

刻画元数据的基本特性是与其他一些资源（例如，数字对象、馆藏，甚至服务）的关涉性（aboutness），即元数据是关于（about）一个特定的资源。例如，都柏林核心（Dublin Core）元数据记录是关于某个特定的电子杂志的文章或网页。此信息通常被刻画为*描述性的*（descriptive）或*结构性的*（structural），在某些情况下也有*管理性的*（administrative），例如有关知识产权的信息。

由于 6.2 节已经涵盖了诸如 MARC 等一些元数据标准，因此这里将只讨论与元数据有关的计划，这对数字图书馆特别重要。

MARC 等复杂标准用来描述基于互联网的资源。都柏林核心元数据标准是针对与这些标准相关联的复杂性和成本而开发的。该标准定义了 15 项元素来描述任何类型的数字对象：7 项描述内容（名称、主题、说明、来源、语言、相关资源和范围），4 项处理知识产权问题（创作者、出版者、发行者和版权），其他 4 项处理数字对象的实例/表现等属性（标识、数据、类型和格式）。合格的标准版本可以对 15 项原始元素进行改进，使其含义更窄或限制更多。为了解释改进的元素值，定义了编码模式。这些模式包括受控词表、形式符号或分析规则。强制使用都柏林核心来描述资源库中的资源，是开放档案计划元数据获取协议（Open Archives Initiative Protocol for Metadata Harvesting）所实现的互操作性基础之一（见 17.4.3 节）。 716

元数据编码和传输标准（Metadata Encoding and Transmission Standard，METS）[1025] 是一个对与数字图书馆中的对象有关的描述性、结构性和管理性元数据进行编码的开放标准，采用万维网联盟（World Wide Web Consortium，W3C）的 XML 模式语言表示 [1659]。该标准由美国国会图书馆维护，正在由数字图书馆联盟（Digital Library Federation）作为协议开发 [710]。METS 框架支持自由选择元数据格式，避免数据重复。图 17-2（摘自 [287]）显示了一个简单的 METS 文档例子，它由两章组成，每一章存储在一个 HTML 文件中，并链接到一个中央 HTML 文件。每章保存一些预留的元数据，而文档级的元数据存储为都柏林核心格式。METS 文档最多包含 7 节（section），其中 5 个如图 17-2 所示：METS Header（METS 头）、Administrative Metadata（管理性元数据）、Descriptive Metadata（描述性元数据）、(File Section) 文件节和 Structural Links（结构链接）。其中结构化内容是唯一必需的节，其余节的内容则映射到已编码的结构中。为了清楚起见，这个例子只显示了节之间的连接、描述性元数据和文件。

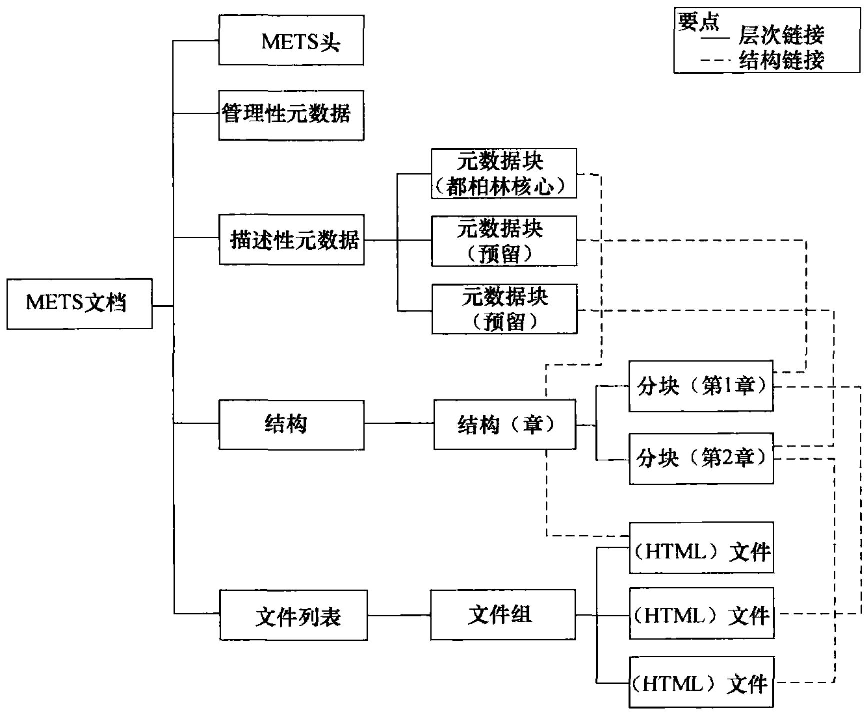


图 17-2 由两章组成的 METS 文档示例

MODS 元数据对象描述框架是基于 MARC 的书目元素集框架，可用于多种用途，特别是图书馆应用 [1026]。MODS 允许从现有的 MARC 21 记录中选择数据，并允许创建原始资源的描述记录。它包括 MARC 字段的一个子集，并使用基于语言的标签，而不是数字标签。在某些情况下，它从 MARC 21 书目格式中重新组合元素。

元数据不仅限于数字对象。为达到以下目的，可以使用馆藏元数据 [765，1094]：

- 利用能够提供馆藏信息访问的搜索客户端软件进行馆藏注册。
- 将馆藏信息提供给网络代理，促进网络发现。
- 编制文档。
- 管理，例如，为广泛的馆藏信息指定存储中心点。

使用馆藏级的元数据，可以为获取自不同甚至异构供应者的项级的元数据记录提供有价
717 值的环境信息，从而提高搜索和发现任务的整体质量 [575]。

W3C 的 Web 服务描述语言（Web Service Description Language，WSDL）根据服务能够理解的词汇和能够处理的消息，提供描述 Web 服务的机制，同时提供特定的、依赖于协议细节的描述机制，用户必须遵从上述机制在确定的服务端点访问服务 [465]。

元数据获取通常是昂贵和费时的，尤其当有大量的信息需要加以描述时。自动元数据提取是一个发展中的研究领域。Han 等人介绍了基于支持向量机的分类方法，从研究论文的标题中提取元数据，该方法在相同的任务上优于其他的机器学习方法 [694]。MetaExtract 是基于都柏林核心＋（Dublin Core plus）的教育元数据自动标注系统，它使用自然语言处理技术来处理教育文档 [1750]。在 [784] 中，作者着重研究从普通文档（例如，演讲、书的章节、技术论文、说明书、报告和信函等）提取标题的方法。Cortez 等人 [430] 提出了一种基于知识的无监督方法，以帮助在任何给定的格式中提取出正确的引用成分（例如，作者、标题、会议地点和页码）。Paynter [1250] 重点研究元数据自动标引工具的评价，并讨
718 论了其优点和局限性。

17.4.3 资源库/档案库

资源库用于存储数字对象的馆藏，并提供基于句柄存储和检索特定对象的基本方法。此外，在许多情况下，它们还提供了诸如安全性 [68] 等附加的特性，最重要的是，提供远程和分布式资源库访问协议 [961]。

一种重要的资源库是所谓的*机构资源库*（institutional repository）。机构资源库负责存储和归档特定机构或机构联盟的完整知识产品，供长期保存、访问和分发 [459，1066]。这里的机构在大多数情况下是教育机构（例如大学）。在 [535] 中可以发现一个机构资源库的综合列表。

接下来，我们讨论与资源库/档案库相关的重要问题，包括互操作性和保存等相关方面。

1. 互操作性

内容分发是数字图书馆的关键功能之一，但它也带来一些最困难的挑战：互操作性（interoperability），即多个数字图书馆系统共同工作来实现分布式搜索或浏览等共同目标的能力。传统上，数字图书馆的互操作方法因每种数字图书馆组件所需要的标准化工作量不同而有所不同。下面我们介绍一些这样的方法。

(1) 联合服务

根据这种互操作方法，有些组织根据事先认可的规范构建其服务，通常是从正式标准中选择规范。每个组织必须执行当前所有的协议，并保持更新，这样就形成了联合工作。联合

工作的一个很好的例子是那些使用 Z39.50 协议的工作。Z39.50 是一个综合协议，允许检索系统之间按客户机/服务器方式相互通信 [1146，62]。该协议定义了查询语言、记录语法和一种称为 EXPLAIN 的工具，该工具允许客户机获取服务器软件的功能信息，和存储在服务器上的每个数据库中信息特点有关的各种信息。

Z39.50 的实施者提出了相关的新计划，以使 Z39.50 支持更广泛的知识/语义内容。这些新计划的目标是降低实施障碍，同时保留 Z39.50 现有的知识贡献。其中的两个计划是 SRU/SRW 伙伴计划 [1218]。搜索和检索 Web 服务（Search & Retrieve Web Service，SRW）计划是国际合作的一部分，以开发一个基于 Web 的标准文本搜索界面。它在很大程度上借鉴了 Z39.50 的抽象模型和功能，但删除了许多复杂性。SRW 使用常见的 Web 开发工具构建（WSDL、SOAP、HTTP 和 XML 等，参见第 6 章）。搜索和检索 URL 服务（Search & Retrieve URL Service，SRU）是一个基于 URL 的 SRW 替代方案。它通过 HTTP 使用 GET 方式发送消息，并且把 SRW SOAP 的请求组件映射为简单的 HTTP 参数。 719
移走 SOAP 包装器后，对 SRU 请求的响应和 SRW 请求相同。

特别地，在数字图书馆领域，Dienst 协议 [961，475] 是联网计算机科学技术参考图书馆（Networked Computer Science Technical Reference Library，NCSTRL）项目的核心 [957]。Dienst 基于 HTTP 协议，并引领了轻量级协议的流行趋势，这是该领域未来的重要基石。它只包含了少数几个所谓的"动词"（例如，Search、Fetch），用于服务之间的通信。

（2）获取

联合方法有一个固有的问题：性能和可靠性是联合服务（federated service）最薄弱的环节 [1299]。除此之外，创建大型联合需要为实现标准和协议，并与任何变化保持同步付出代价。这些困难激发了最近一些旨在创造更松散的数字图书馆组合的工作。其基本概念是，参与者付出很小的努力，实现一些基本的共享服务，而不规定完整的一套协议。开放档案计划（Open Archives Initiative，OAI）是说明这一点最好的例子，它促进了都柏林核心成为标准的元数据格式，并且定义了一个简单的标准元数据获取协议。运行该服务的数字图书馆可以从实现该协议的其他数字图书馆获取元数据，并存储到自己的中央资源库中。

特别对 OAI 而言（见下文），最初阻碍其实现的是某些档案库，因为需要用到少量的编码和中间件层建设。在本地资源库较小的情况下更是如此。在其他情况下，有些资源库有时不能很好地匹配 OAI 架构，例如那些基于 Z39.50 协议的资源库。此外，非常小的档案库可能缺乏人力资源来安装和维护服务器。而且，有些档案库不会采取任何积极的措施开放所有的内容。在这种情况下，我们现在要讨论的收集就成为唯一的选择。

（3）收集

如果各组织不准备以任何正式的方式进行合作，那么通过收集（gathering）可公开访问的信息仍然可能实现基本的互操作性。收集的最佳例子是通过 Web 爬取实现的（见第 12 章）。因为收集有最小的员工成本，所以可以为大量的数字图书馆提供服务，但与那些由伙伴图书馆充分合作所实现的服务相比，质量相对较差。这主要是由于可收集到的数据存在质量问题，缺乏结构和出处等信息。

（4）开放档案计划

开放档案计划（Open Archives Initiative，OAI）及其元数据获取协议被许多人认为是过去十年中数字图书馆领域最重要的发展之一。因此值得更详细的介绍。两者的根源是数据和服务提供者的清晰分离。数据提供者使用 OAI 技术框架作为表示关于内容的元数据的一 720
种手段。服务提供者使用 OAI 协议从数据提供者获取元数据，并使用元数据作为开发增值

服务的基础［963］。OAI 技术框架解决了两个著名的元数据需求：互操作性和可扩展性（或社区特异性）。元数据的互操作性要求所有 OAI 数据提供者使用共同的格式提供元数据：无修饰的都柏林核心元数据集[⊖]（Unqualified Dublin Core Metadata Set）。在 OAI 技术框架内，社区特异性的描述，或元数据的特异性，通过支持并行的元数据集解决。OAI 技术框架允许两种类型的选择性获取：1）按日期，基于记录的日期戳[⊜]，请求可能包含需要获取的元数据的日期范围，这可能是完全的（两个日期之间）或部分的（只有下界或者上界）；2）按集合，可能是分层组织的可选项目组。OAI 基础结构并不指定集合的实际含义。OAI 元数据获取协议（Open Archives Initiative Protocol for Metadata Harvesting，OAI-PMH）包含 6 个动词：

- GetRecord：用于从资源库的一个项目中检索个别（元数据）记录。需要的参数包括所请求记录的标识符（或句柄），以及应在记录中包含的元数据的格式。
- Identify：用于检索资源库的有关信息，包括：1）人工可读的资源库名称；2）资源库的基本 URL；3）资源库支持的 OAI 协议版本；4）资源库管理员的电子邮件地址。
- ListIdentifiers：用于检索从资源库获取的记录的标识符。可选参数允许选择标识符，其依据为标识符在资源库的一个特定集合中的隶属关系，或者在特定日期范围内修改、创建或删除标识符。
- ListMetadataFormats：用于检索资源库可用的元数据格式。可选参数约束了对一个特定记录可用的元数据格式的请求。
- ListRecords：用于从资源库中获取记录。可以使用与 ListIdentifiers 动词相同的选择标准。
- ListSets：用于检索资源库的集合结构。

2. 保存和归档

软、硬件的过时和计算机媒质的退化，将给所有类型的数字内容带来风险，在不那么遥远的将来，可能无法给对这些内容感兴趣的用户提供服务。随着越来越多的数字信息的出现，对
721 什么样的信息应予以保存，和它该如何以经济有效的方式保存，都必须引起注意［1338］。

以美国为例，美国国会图书馆领导了国家数字信息基础设施和保存计划（National Digital Information Infrastructure and Preservation Program）［1024，999］，这是一个由联邦和非联邦实体组成的联盟，其创建目的是确定数字资源保存的政策、协议和战略。与此相一致，美国国家科学基金会，依据国家科学基金会/国会图书馆数字归档和长期保存联合（Digital Archiving and Long-Term Preservation，DIGARCH）计划，资助了第一个与保存和归档相关的项目。

除了法律、社会和经济影响外，社会上的普遍看法也反映在文献中，认为资源库和档案库有责任保存它们持有的内容［1386，1362］。此外，这些档案库应该保证其信息真实可靠，明白易懂［627］。它们应该透明地公开其程序和做法，以使公众相信档案库已经得到了妥善的保存。

从历史上看，对于数字资源保存，已经提出了五种主要的技术方法：

1）迁移（Migration）：从一种数字格式转化为另一种格式，后者通常是前者的直接后

⊖ 所有字段都认为是可选的。

⊜ 定义为“项目的创建、删除或最新修改日期，其作用是从该项目传播记录元数据的变化”。

继（例如，从 JPEG 到 JPEG 2000）[442]。

2）仿真（Emulation）：通过保存原始程序或者创建可以仿真旧环境的新程序，重建原来的操作环境 [1387]。

3）包装（Wrapping）：将需要保管的对象用足够的人工可读的元数据包装起来，使其在未来能够解码 [1671]。这个想法经过了 Gladney 的一系列探索，他创造了术语“可信数字对象”（trustworthy digital objects）[626，628，627]：这种数字对象采用元数据描述其来源，用加密方式封装，其公钥信任机制源于受尊敬的某些机构，并提供了管理信息标识符的具体方式。

4）刷新（Refreshing）：从一个位置复制位流到另一个位置，不管物理介质是否相同 [1013]。

5）复制（Replication）：制作足够的数据副本。这是 LOCKSS 项目 [1079] 所倡导的主要方法。在世界范围的测试中，LOCKSS 项目已经开发并部署了一个保管数字资料的对等（peer to peer）系统。它由大量独立的、低成本的、持久的 Web 缓存组成，通过使用复杂的投票模式，合作来检测和修复破坏的内容。

注意，这里不考虑存储对象的物理介质本身的劣化，因为这是介质本身的特性，而不是对象的。但是，由于刷新是常用的方法，我们承认这是一个重要问题。由于成本、操作和技术上的原因，迁移是前三种技术中最广泛使用的 [1671]。但是，理想的解决方案应该是所有技术的结合 [818，1671]。已经被应用的一个例子就是基于通用虚拟计算机（Universal Virtual Computer，UVC）[628，1049] 的组合。数字对象的归档方法包括规定数据（存储在物理介质）所需的处理过程，以便向未来的客户端返回信息。流程规范和逻辑视图定义都与数据一起存档（包装）。程序行为的归档方法是保存原始的可执行对象代码，和原始计算机执行每条机器指令需要的处理规范（仿真）。在这两种情况下，处理规范都是基于 UVC 的，该 UVC 是一般化的，但在未来基本够用。 722

空间数据系统咨询委员会（Consultative Committee for Space Data Systems）提出的开放档案信息系统（Open Archival Information System，OAIS）参考模型，提供了一个进行数字化保存的研究和实验参考架构 [417]。OAIS 描述了通用数字资源库的功能：数字对象如何按需要准备，提交给档案库，长时间保存，维护和按需检索。OAIS 的优点在于，建立了描述资源库体系结构和比较实现的公共术语和概念，其中比较实现是指在不指定组织结构应该采用哪种特定实现方式下进行比较。它也没有涉及具体技术、存档技术或者内容类型。

17.4.4　服务

服务是访问数字图书馆的主要渠道。通过服务可以创建、发现、丰富和访问信息，并最终在数字图书馆中使用。数字图书馆服务可以从许多方面刻画和理解。一种方式是分析用户（读者、参与者）或其他服务的输入（所消费的信息）和输出（所产生的信息）。这是很重要的，尤其是在服务只能通过“黑盒”模式访问，而不知道其内部过程或组件的情况下。这种分析有助于理解服务是如何使用的，有哪些要求，边界在哪里。

[642] 全面地研究了数字图书馆的服务行为。该分析的结果是，定义了服务的分类词典，如表 17-1 所示。分类词典是通过对服务的输入和输出及其共性进行深入分析而获得的。具有类似 I/O 行为的服务被组合在一起。在分类词典的最高级，服务分为*基础*（Infrastructure）和*信息满足服务*（Information Satisfaction Services）。后者区别于前者的地方在于，其主要功能需要表示个人物品或兴趣特点（例如，用户轮廓、用户认为相关的文档集）的用户输入，或者信息需求的显式表示（查询、锚点、期望对象的句柄等）。

表 17-1 数字图书馆服务/活动的分类词典

基础（Infrastructure Services）			信息满足服务（Information Satisfaction Services）
资源库建设（Repository-Building）		增值服务（Add-Value）	
创建型（Creational）	保存型（Preservational）		
获得	保护	注释	绑定
创作	转换	分类	浏览
编目	复制	聚类	定制
爬取（聚焦）	迁移	评价	传播
描述	翻译（格式）	提取	扩展（查询）
数字化		索引	过滤
获取		链接	推荐
提交		日志	请求
		度量	搜索
		分级	
		（同行）评议	
		综述	
		训练（分类器）	
		翻译（语言/格式）	
		可视化	

基础服务负责生成数字图书馆的组件，这些组件是开发信息满足服务的基石。基础服务又细分为*资源库建设*（Repository-Building）和*增值服务*（Add-Value service）。资源库建设服务负责生成“最小数字图书馆”的基本构建块，包括数字对象、元数据规范、馆藏和元数据目录。增值服务汇总投入的总价值（例如，注释、评价、结构化、索引、度量、日志、分级、评议、翻译和可视化），或者将对象连接在一起（例如，通过训练和分类、聚类、索引和链接）。

可用性问题是一个与服务有关的、却未在数字图书馆文献中充分体现的重要方面[1204]，即参与者使用数字图书馆服务的容易程度，主要通过其用户界面体现。有些受控的、基于实验室的早期数字图书馆可用性研究有［578，1786］。［899］对 ACM、IEEE、NCSTRL 和其他一些数字图书馆的用户界面进行了对比和评价，并发现了许多问题。其他的研究主要针对教育环境，集中在信息满足服务的可用性问题，即读者查找、使用，或者与从数字图书馆中发现的资料进行交互的方法（如［207，208］）。例如，Borgman 等人［235，236］研究了亚历山大数字地球原型（Alexandria Digital Earth ProtoType，ADEPT）的设计并加以评价。ADEPT 是有关地理参照信息资源的数字图书馆，用于本科教学。访谈表明，该项目的地理资源能更好地服务于研究，而不是教育目的，这意味着需要为教学设计新的服务。[1110] 进行了网上调查，采访了使用 NEEDS（美国国家工程教育数字图书馆）的教育工作者和学生。结果发现，数字图书馆需要开发和实现更有效的服务吸引读者，这对它们的生存至关重要。例如，调查发现了对评论、服务反馈和网上讨论的需求。另一项研究针对焦点小组（focus groups）进行了访谈[⊖]，访问了有着不同教育背景的在职教师、职前教师和科学图书馆馆员，了解他们对数字图书馆服务质量的看法［1544］。Adams 和 Blandford 介绍了一项有趣的研究，通过访谈、焦点小组和对 150 个用户进行观察，并对结果进行深入分析，研究了数字图书馆如何支持健康和学术领域有关用户的“信息之旅”[10]。

有时显式的用户研究很难进行，这时的替代方法是*日志分析*（log analysis）。日志分析可以作为了解读者如何实际使用数字图书馆的系统和服务，以及这些技术如何支持用户的信

⊖ 一种定性评价形式，邀请一群人在一起，针对一个产品、概念或技术自由发言，说明他们的态度或感觉。

息搜索活动的主要来源。日志记录和分析有助于评估系统，为改善和增强新的服务创造机会［851］。已经提出了基于 XML 的数字图书馆日志建议标准，它比传统的 Web 服务器日志承载了更多的信息［639，645］。除传统研究之外，日志分析在数字图书馆中的其他创新性使用还包括导出加权的期刊关系网络，确定研究趋势［223］，利用 OAI-PMH 获取的日志条目来开发推荐系统［1621］等。

17.5　社会经济问题

17.5.1　社会问题

数字图书馆领域最困难的挑战不是技术，而是涉及社会经济的问题和实践。在数字图书馆所涉及的社会问题中，最突出的包括：合作与协作（包括信息共享）方面；社交网络；文化遗产和保存；数字鸿沟和国际化。

如前所述，数字图书馆有可能彻底改变传统的信息链，成为用户与信息和用户之间交互的主渠道。这将打开很多合作/协作的机会。举例来说，伙伴关系评议模型（Partnership Review Model）针对学术和复杂的数字教育资源（如课程模块、模拟和数据分析工具），探索合作和无协调的同行评议［1672］。内容协作生产（在某些情况下就是竞争力）的最佳例证是 Wikipedia 项目［1689］，其他项目，如 PlanetMath［940］也证明了这一点，其关键是通过探索由志愿者组织和社区规范的知识生产，保证长期项目的可持续性。这里，志愿者社区以平等的方式为项目组件做出贡献，没有传统的层次结构的组织或经济补偿，而是存在一些流程将它们结合起来，产生一个统一的工作。这种形式的内容协作生产也称为共同对等生产（Commons-Based Peer Production，CBPP）。因为互联网已经降低了某些沟通和合作障碍，CBPP 已经出现并蓬勃发展，成为一种可行的完成大型复杂项目的替代方式［182］。

近来，Web 上出现了一个有趣的社会现象——*社交网络*（social networks），即由合作者、同事或朋友组成的、有着共同兴趣的非正式社区［881］。换言之，社交网络通过显式或隐式的兴趣爱好，以及相互交流或相互关系，将人们连接在一起［1260］，最好的例证是 Facebook［540］、MySpace［1168］、Orkut［1234］和 Friendster［596］等关系网站。有些数字图书馆项目已经探索了社交网络的应用，这些应用进行了不同形式的协同，例如通过探究合作关系对科学文章的作者进行消歧［695］，提供推荐服务［790，791，1259］，并将数字图书馆整合到特定的社区网络［310］等，然而，这些丰富的社会化结构的潜力还有待于进一步挖掘。 725

我们已经谈到了保存问题的一些技术方面。不过，其中所涉及的社会问题也许更难解决。首要挑战就是要建立起公众意识，即如果在信息生命周期里不考虑保存问题，那么就会有巨大的危险，其后果是失去目前已经生产的和正在生产的大多数（数字）知识和文化遗产。从意识到这一点开始，到建立保存文化使得保存工作嵌入知识的生产和消费链中，都是巨大的挑战，这些挑战包括：1）确定责任；2）改变既定做法和方法；3）寻找资源来处理额外开销；4）确保所保存信息的真实性［627］等。

17.5.2　经济问题

数字图书馆的经济问题与多个方面相关：安全（例如，授权、验证和水印等）；法律（例如，条款和条件、专利、商标、版权、知识产权和数字版权管理）；出版（例如，Eprints、自归档、编目成本和开放馆藏）；可持续性。可持续发展最有效的方法是使用之前讨论的协作手

段。在上述问题中，我们将讨论三个重要的问题：版权管理、访问控制和出版问题。

1. 版权管理

知识产权包括法律和技术系统，保护以产品形式表达的想法或知识独家使用的个人权益。一种类型的知识产权是版权，即复制原创作品的权利，保护期限通常是有限的。传统的版权管理得益于资料的物质形体 [802]。随着从传统时代迁移到数字时代，数字资料的复制和传播变得非常容易，这就对保护这些权利带来了许多挑战。对保存在数字图书馆和资源库的受保护资料更是如此。数字版权管理（Digital Rights Management，DRM） [326，802] 是一个术语，它不仅包括安全和加密问题，也包括说明、鉴定、交易、保护、监测和跟踪数字资料的一切权利。Ianella [802] 给出了 DRM 架构的概述。

2. 访问控制

对知识产权管理的忧虑带来了许多数字图书馆实现某种类型访问控制的需求。例如，美国国家科学数字图书馆（U. S. National Science Digital Library）的核心架构（参见 17.7.2 节）使用了一些协议和系统（如 Kerberos [1533] 或 Shibboleth [1191]），将身份验证和成员授权分配给不同用户社区的管理员 [959]，并利用权限管理代理（Rights Management Broker），根据图书馆中的项目（由元数据指定）和用户（从授权服务获得）的性质决定是
726 否允许用户访问。其他系统则直接在系统架构（例如， [1562]）或内部数据模型 [283，871] 中纳入身份验证和访问控制。

3. 出版问题

鉴于不能正确支持科学界的需求，目前的科学交流过程中已在过去几十年受到了严厉的批评 [908]。许多因素导致了这一点，包括：1）出版商对期刊许可权的提价速度比通胀快很多，研究界和大学图书馆的预算却不高，造成了失衡现象；2）由于版权转让给了出版商，作者无法改进以及与同行分享自己的工作，并因此获得必要的科学确认；3）提交文献和实际出版时间的巨大延迟。

在此背景下，数字图书馆和资源库已成为解决上述问题的有效替代方式。许多数字图书馆可以作为预印本（pre-prints，科学文章出版前的版本）的中央存储库，以方便其流通，从而使作者得到更多的及时反馈。这一问题也导致了“开放存取” （Open Access）运动 [706] 的发展，以及后继的开放档案计划等提高互操作性的重要活动 [963]。

由于经济的可持续发展，许多数字图书馆都支持“自存档”服务，使研究人员能够轻松地将自己的工作归档，并与同行分享成果和著作，通过让感兴趣的社区参与以节省成本。事实上，根据 Harnad [706] 的阐述，自存档的目标是让科学论文的全文可见，让任何感兴趣的用户通过互联网进行访问、检索和使用。使用“自存档”这个术语，是因为由作者本人负责在资源库中插入自己的工作，并填写任何相关的元数据描述。然而，只要作者允许，自存档也可以由第三方进行。事实上，目标社区的积极参与是一切数字图书馆成功的关键。对于目标社区的自存档界面，已经有了评测报告，例如 [1631]。这种自存档和不受限的联机资料发布已由布达佩斯开放存取计划（Budapest Open Access Initiative）等国际运动推广 [216]。

17.6 软件系统

以下是支持数字图书馆创建的软件系统。它们大多可以自由下载并安装在本地，由目标社区对许多数字图书馆的建设提供支持。当前许多常用的数字图书馆软件系统或者作为研究原型开发，或者与企业合作开发。这里，我们将介绍以下软件系统：Greenstone、Fedora、
727 Eprints、Dspace、ODL 和 5S 套件。

17.6.1 Greenstone

由怀卡托大学计算机科学系开发的 Greenstone 系统受到了联合国教科文组织（UNESCO）的支持，是最早的综合性数字图书馆软件系统之一，目前仍然非常受欢迎。Greenstone 的重点是资料的出版。其架构围绕插件和分类器 [1701，1702]。插件是负责“识别”具体数据格式，并将其转换为内部表示的小软件模块。现在有针对标准文件格式（如 PDF、HTML、Microsoft Word）和多种元数据格式的插件。此外，新的插件可开发和添加到系统中，以处理新的格式。Greenstone 分类器不同于第 8 章中所述的分类任务，它用于从元数据建立浏览索引，例如按字母顺序的列表（如标题）、日期选择列表和层次浏览结构等。与插件一样，也可为特殊用途的浏览结构编写新的分类器。系统还为收集和丰富数字图书馆馆藏的流程提供便利的工具 [128]，支持全文和基于元数据的搜索，使用压缩技术以最大限度利用存储资源 [1709]。该系统最近已重新设计，以实现 OAI-PMH 和 METS 等新的要求和标准 [287]。

17.6.2 Eprints

Eprints [1666] 由南安普敦大学开发，旨在促进科研材料的自归档，以提高学术著作的知名度和影响力。它是实现机构资源库的领先软件，在世界各地有庞大和不断增长的安装基地。

Eprints 由服务团队提供支持 [1567]。团队提供收费的建议和咨询，包括最初的帮助和指导、员工培训和软件定制，以满足特定机构的需求。

17.6.3 DSpace

DSpace [1562] 是一个开源软件系统，由麻省理工学院图书馆和惠普合作开发。该系统创建的目的是针对研究和教育类数字化资料的长期保存，开发并部署解决方案。公开源代码的目的是围绕 DSpace 形成一个开源社区。这些 DSpace 社区负责添加特色、改进系统功能，以及为满足特定机构的要求和需要改编系统。其内部术语和数据模型的基础是开放档案信息系统（Open Archival Information System，OAIS）参考模型所提出的概念 [417]。该系统的另一个强有力的组成部分是支持数据提交的工作流子系统，其策略可根据不同的社区和馆藏而调整。 728

17.6.4 Fedora

灵活可扩展的数字对象资源库架构（Flexible Extensible Digital Object Repository Architecture，Fedora）是一个基于 Java 的开源资源库框架，由美国康奈尔大学信息科学系与弗吉尼亚大学图书馆联合开发。Fedora 为复杂数字对象及其相互关系的存储、管理和分发而专门开发 [962]。这些对象除了绑定的服务以外，可能还含有本地或分布式的内容。允许一个对象有多种表示形式（称为分发），其中甚至有可能是动态创建的。采用资源描述框架（见 6.4.4 节）表示对象之间的关系，并进行存储，以供进一步查询。Fedora 架构通过 Web 服务实现，为其他应用（例如，数字图书馆）的创建提供基础。

Fedora 的对象模型支持多种复杂的对象类型，如文档、图像、电子图书、多媒体对象和软件等。此外，这些类型可以结合起来创建新的对象，并有可能将服务绑定到对象以产生动态内容。该模型还支持对象之间的关系，以便在资源库中的对象之间可以有一些语义关系

（例如书和它的各章，或会议及其论文集）。Fedora 的对象模型也可以看做是一个由内部弧（将数字对象联系到其表达形式）和外部弧（数字对象之间）组成的有向图。Fedora 提供了*资源索引*（Resource Index）服务，以便对这张图进行图形的存储和查询。

17.6.5 ODL

开放数字图书馆（Open Digital Libraries，ODL）是第一个背离数字图书馆软件系统整体性视角的框架，它主张数字图书馆向组件化（componentized）方向发展。数字图书馆的功能通过多个软件组件实现，这样就允许重用性和模块化。发展的重点已经从代码实现转变为组件集成，再变为对具体要求的自适应性。

在 ODL 框架内，广受欢迎的服务，如搜索、浏览和注释，被定义为自包含的组件。对于每个组件，ODL 指定用于与数据提供者和其他对等组件交互的接口。其中每个组件实现为一个开放档案（Open Archive），使用 OAI-PMH 共享数据（见 17.4.3 节）。因此，ODL 组件可以作为数据或服务提供者工作，也可以同时承担这两种角色。组件间的通信通过 OAI-PMH 的一个扩展版本实现。大多数扩展具有 OAI-PMH 不包括的共同特征，因为 OAI-PMH 的重点是元数据获取。该框架包含的一个重要功能是一种将记录添加到档案库的机制，虽然不一定要通过 OAI-PMH 输出数据，但组件间的互动是必不可少的。

通过在松散的底层协议框架之上分层堆积特定的语义，各个组件 ODL 协议具体化了 OAI-PMH，成为 OAI-PMH 的详细描述。例如，集合（set）参数在 OAI-PMH 内没有规定任何特定的含义，因此可以很容易地用来表示记录选择的又一个标准。在这种情况下，对于
729 支持搜索引擎组件的 ODL 协议，查询作为一种选择（和排序）标准被嵌入到集合参数中。

在重载搜索组件的语义后，有这样一个 OAI 请求的例子：

```
verb=ListIdentifiers\&set=odlsearch1/computer\%20science/1/10
```

对这个查询的响应应当包含与“computer science”这个查询最相关的前 10 个文档的标识符。

类似地，为其他服务也定制了 ODL 协议。ODL 已应用于许多数字图书馆项目，其实用性、可扩展性和可重用性得到了证明。然而，该框架所呈现的可扩展性问题，以及所使用的 OAI-PHM 专有扩展对互操作性造成了一定影响。

17.6.6 5S 套件

不同于其他大多数项目，5S 工具套件采用模型驱动的方法来创建和生成数字图书馆应用。这些工具围绕 5S 数字图书馆形式化框架而组织 [643]，如图 17-3 所示。形式化框架是开发元模型（meta-model）或数字图书馆构建语言的基础 [640]。形式化概念通过 XML 语言的结构具体化，这种语言被数字图书馆设计者用于对特定和定制的数字图书馆进行结构建模。与 UML 等通用建模语言相比，和领域更接近的建模语言称为*领域特定语言*（Domain Specific Language，DSL）[1623]。

直接用 XML 构造复杂模型是一项艰巨的任务，可能产生很难理解和解释的大文件。为了方便数字图书馆建模，开发了 5SGraph 图形化建模工具 [1786]。5SGraph 允许数字图书馆的设计者用图形方式建立数字图书馆模型，并以 5SL 的形式输出模型。该工具经可用性测试，在满意度、舒适性、正确性、所生成模型的完整性，以及易学性等方面，都产生了非常好的效果 [1786]。由此产生的 5SL 代码是另一种工具 5SGen 的输入。除 5SL 代码之外，5SGen 还使用了一个基于 Java 的 ODL 组件包装器，产生数字图书馆的运行版本。有趣的

是，该工具能够为一项服务以连贯的全局行为整合多个场景规范。但是，5S 套件的缺点在于，它是为了表明在数字图书馆构建上的创新思路而开发的研究原型，不具备鲁棒性，也不支持其他系统。

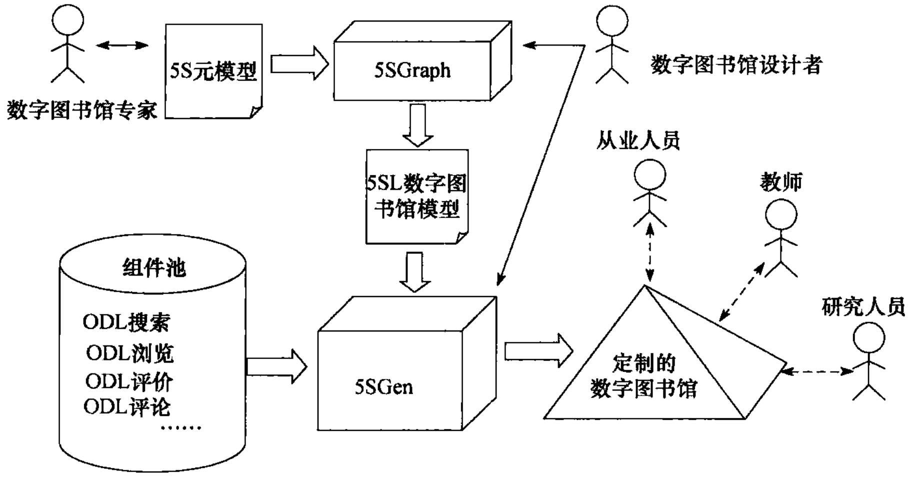


图 17-3　5S 套件

5S 套件最近增加的功能包括：向导工具，它支持受控的工作流，以配置组件化的数字图书馆 [1423]；评估数字图书馆质量的工具 5SQual [956]；支持 DSpace 配置和定制的 5SL 版本 [658]；新的组件池 WS-ODL，使用 SOAP 等 Web 服务协议进行组件的相互通信，并改进了 ODL 框架的一些可扩展性和效率问题 [1363]。 730

17.7　数字图书馆案例研究

今天存在着数以百计的数字图书馆项目。这里，我们将列举一些有趣的例子。

17.7.1　联网学位论文数字图书馆

联网学位论文数字图书馆（Networked Digital Library of Theses and Dissertations，NDLTD）[1190，1542] 是一个聚焦于电子学位论文的数字图书馆全球网络。虽然当初没有中央控制机构，几乎也没有预算，但 NDLTD 却是一个非常成功的数字图书馆案例。

为了促进可持续性，需要吸收全世界尽可能多的机构参与，向他们展示参与这个过程的所有机构将以某种方式从中受益。由于不需要归档学位论文的副本，大学图书馆将看到由此能够产生巨大的经济效益。研究生院可以通过电子工作流来简化和降低提交和处理电子学位论文等活动的费用。教授和学生也可以使他们的工作更加受到关注。学生还可以学习到更多的有关电子文档生产的知识。这些参与机构要求学生对自己的工作进行编目，掌握这些知识将有利于他们今后的学术生涯。参与机构也将打开大门，使普通公众了解更多的知识。

截至 2009 年 10 月，NDLTD 在五大洲有着近 100 家正式成员⊖。联合国教科文组织也支持这项计划，并资助了电子学位论文创建指南，以帮助各机构设立自己的电子学位论文项目。NDLTD 的联合目录包含近 80 万条记录。在目录上建立了搜索、浏览和聚类服务。这 731

⊖ 指的是每年交纳年费的成员。实际上 NDLTD 的合作单位有 250 多家大学。

一计划也推广了他们自己的电子学位论文元数据编目和共享标准。

17.7.2 国家科学数字图书馆

国家科学数字图书馆（National Science Digital Library，NSDL）是一个有趣的研究案例，这不仅是因为其雄心勃勃的目标——用他们自己的话说，“为高品质的资源（例如，内容和服务）提供有组织的访问，为各种水平的科学、技术、工程和数学教育提供创新性的教学工具”，而且也与它的重要性有关。该项目由美国国家科学基金会支持，已资助了150多个项目，自成立以来，已投入1亿多美元。NSDL资助的项目主要包括三种类型：1）路径（Pathways）项目，这些项目的目标是提供主要学习者社区所需要的内容管理；2）服务（Services）项目，目标是开发服务以支持用户和资源库供应者，提升图书馆的影响力、效率和价值；3）目标研究（Targeted Research）项目，这些项目的目标是探索可直接适用于数字图书馆的馆藏、服务和其他发展方面的特定主题。核心整合项目有责任以连贯的和可互操作的方式整合所有这些项目[959]。

17.7.3 ETANA-DL考古数字图书馆

ETANA-DL考古数字图书馆（ETANA-DL Archaeological Digital Library）是一个很好的为特定领域（domain-specific）设计的数字图书馆的例子。ETANA-DL作为一个考古数字图书馆，旨在收集、记录、集成和保存在考古调查和发掘中收集的考古资料。当前版本的ETANA-DL从中东和中东以外的多个考古遗址汇集数据。这些数据包括雕像的照片、与出土的种子和骨骼有关的信息、探方（excavated unit）的图纸和地层图等。ETANA-DL也为考古学家和普通公众提供了推荐和基于内容的图像检索[1635]等服务。

ETANA-DL有一点很有趣：它在开发中使用了理论和模型驱动的方法。它扩充了原来的（最小）5S框架，以覆盖考古学领域的具体概念和数字图书馆集成问题[1458]（见17.8.2节）。为了应对增强的框架，5S工具套件也进行了扩充，以便部署在所创建的第一个原型上。例如，在全局（整个数字图书馆）和本地数据（个别发掘遗址）之间进行模式转换的映射工具已被纳入该套件。这些模型作为扩展5S考古元模型的实例，使用5SGraph创建[1786]。ETANA-DL的经验可用于其他特定领域数字图书馆的创建（例如，生态/生物
732 多样性[473]和教育[388]等领域）。

17.8 趋势和研究问题

数字图书馆领域的许多研究挑战已经在前面的章节中讨论过。这些挑战包括但不限于：1）信息（主要是元数据）提取；2）（复杂）数字对象的新模型；3）资源库管理和可扩展性；4）互操作性；5）数字化保存；6）社会经济问题；7）数字图书馆架构等。在本节中，我们将集中于两个具体的挑战：数字图书馆的评价和集成，并提及其他一些先前未曾讨论的研究挑战。

17.8.1 评价

什么样的数字图书馆是“好”的？正如Fuhr等人[604]所指出的，这个问题的答案取决于你问的人是谁。数字图书馆的评价是困难的，因为这样的评价涉及很多方面，目前还缺乏对于如何在评价中综合考虑所有这些方面的共识。Saracevic[1425]是最早考虑这个问题的人之一。据他分析，对于数字图书馆评价的各种准则、测度和方法论，目前还没有达成

一致意见。Marchionini[1083] 认为，评价的研究既应该是纵向的，以获取更丰富更可靠的数据集，从而进行分析；也应该是多方面的，需要利用各种方法的组合。Fuhr 等人 [604] 提出了一个四维的数字图书馆描述框架：数据/馆藏、系统/技术、用户和用法。[613] 列出了这个领域的部分领先成果，也包括对现有评价过程的分类。

Gonçalves 等人 [644] 认为，以质量为中心的观点可以回答在本节开头提出的问题。他们以 5S 理论为基础，提出了数字图书馆质量维度和指标的形式化体系 [643]。对于质量维度，需要考虑可访问性、准确性、完整性、组合性、一致性、连贯性、有效性、效率、可扩展性、针对性、可保存性、相关性、可靠性、可重用性、显著性、相似性和及时性等因素。对于评价指标，可以考虑响应时间（和效率有关）、迁移成本（和可保存性有关），以及服务失败次数（以评估可靠性）。Gonçalves 等人还讨论了他们所提出的数字图书馆质量维度与信息生命周期模型的一个扩充版本之间的关联，该模型在一次研讨会上得到了一致的认可 [233]。Shen[1460] 在 Gonçalves 等人的工作基础上，着重研究用法方面的评价，建立了信息系统和信息搜寻的采纳模型，提出了从各个方面看待数字图书馆成功（DL success）的整体视角。

归根结底，唯一的共识就是，数字图书馆的研究尚处于早期阶段，对数字图书馆本身的广泛接受和部署也才刚刚起步，需要相当长的时间才能对评价方法取得一致 [613]。虽然如此，评价仍是这类系统发展和被接受的关键。

17.8.2 集成

数字图书馆的研究有一个耐人寻味的方面，即在基本技术和大规模集成两个层次都存在着挑战。10 年来，政府和私人资金对数字图书馆研究项目给予了资助，在基本技术层次上取得 733
了重要的成果。但在大规模集成这个层次上的成果则可以说是不太明显。“数字图书馆集成”的概念比数据的互操作性要全面得多，即便如此，这一概念仍然含糊不清，存在着不同的方法，并提出了不同的解决方案。数字图书馆的集成工作主要集中解决三个问题 [1461]：

1）分布性：地理范围广阔。

2）异质性：在技术层次（例如，硬件平台、操作系统和编程语言等）和概念层次（例如，对相同真实世界实体的不同的理解和建模）的区别。

3）自主性：组件是在何种程度上自给自足，而不只是作为较大系统的组成部分。

我们认为，“数字图书馆集成”意味着隐藏分布性和异质性，同时激发可见组件的自主性（至少在一定程度上）。

许多数字图书馆由不同的自治组织自主开发，事先并无计划对自己的数据和功能提供开放且易用的自动访问机制。不能以无缝和透明的方式访问跨数字图书馆的知识，是知识共享的主要障碍。数字图书馆集成的目标是和谐利用各种自治的数字图书馆，从这些“岛屿”获得知识。与解决方案相比，集成的需求是众所周知的 [1461，960]。

数字图书馆可以在信息和服务等不同层次进行集成。集成信息使得分布式异构资源库成为一个整体。集成服务通过更连贯、更易用的接口，隐藏了被集成的数字图书馆的语法和语义区别，为用户提供更全面的数字图书馆资源使用。虽然对数字图书馆集成已有许多努力，但大多数还是零敲碎打的即兴方法。

17.8.3 其他研究挑战

其他艰巨的研究挑战体现在以下领域：

- 数字图书馆参考模型的创建：尽管在过去 10 年中，本领域取得了很大的进展，但在数字图书馆是什么，它们必须提供什么功能方面，并未取得一致意见。缺乏协商一致的参考模型为本领域整体奠定基础，使得已经做出的工作很难进行比较和整合，得到的成果也很难共享和重用，从而放缓了发展步伐。尽管在这方面已有一些初步
734 的努力 [643，489]，但还有很多工作需要做。
- 引用管理是现代化数字图书馆的一个中心环节。引用是衡量特定科学文章和研究报告影响力或意义的根本证据。评价研究人员的工作，确定其是否应该获得晋升或者基金资助，可以利用引用作为评估其能力和影响力的一个重要证据。引用也可以作为信息检索任务的一项辅助手段，如自动文档分类 [313，432]、索引和排序 [988] 和质量评价 [644] 等。在更广泛意义上的引用[⊖]是数字书目及图书馆项目（Digital Bibliography & Library Project，DBLP）[478] 和计算机科学参考文献（Computer Science Bibliography）[460] 等重要项目的基础。数字图书馆中的引用管理涉及这些方面：1）信息提取方法，以便在任何给定的格式下正确提取引文部分；2）数据清洁和纠错，例如，由于同一作者在论著中使用了多个名字，或者多个作者重名，使得论文的作者被错误分配，或者同一研究人员的成果被错误分割，这个问题称为*名字消歧*（name disambiguation）；3）在数据集成或数据输入任务后去除重复记录。这个任务称为*重复记录检测*（record duplicate detection）或*冗余消除*（deduplication）。事实上，由于需要从多个资源库汇集内容，这是一个更广泛的问题，在数字图书馆中很常见。
- 个性化：数字图书馆在本质上是以社区为本的，可进一步支持根据个人的偏好和特性来调整内容和服务。此外，这样的个性化还可以基于其他因素，例如情境、当前的任务、时间、地点、个人和数字图书馆的交互历史，以及其他一系列虽然没有明确给出、但却隐含在互动过程和周边环境之中的因素。需要开发的新技术包括在特定情境捕获用户对于特定资源的注意力，识别和挖掘用户行为模式等。

17.9 文献讨论

该领域的主要书籍是 Lesk 的《Understanding Digital Libraries》[1006]，现已出版了第 2 版；Arms 的《Digital Libraries》[69]，以及 Witten 关于 Greenstone 的书籍《Managing Gigabytes》[1704]。自该领域出现以来，已经出版了一些概述和综述，例如在《Annual Review of Information Science and Technology》发表的两篇综述 [579，162]。《DLib》杂志 [594] 是了解该领域近期发展的一个极好的信息来源。《Communications of the ACM》、《IEEE Computer》和《Information Processing and Management》等期刊都出版过数字图书馆的专辑 [1221，1222，1223，808，813]。该领域最主要的会议是 ACM/IEEE 数字图书馆联席会议（ACM/IEEE Joint Conference on Digital Libraries，JCDL），该会议于 2001 年起由这两个学会单独组织的会议合并而成。另外，还有一些地区和国家级的会议，如欧洲数字图书馆研究和先进技术会议（European Conference on Research and Advanced Technology for Digital Libraries，ECDL）和亚洲数字图书馆国际会议（International Conference on
735 Asian Digital Libraries，ICADL）。

⊖ 此处解释为作者名称、标题、出版地点或年份等与特定文章有关的书目信息。

开源搜索引擎

——与 Christian Midd 合著

A.1 介绍

为了提供网站内容检索，可以使用商业搜索引擎，也可以使用开源搜索引擎。对于绝大多数网站而言，使用商业搜索引擎不是一个可行的选择，或者因为经济原因，或者因为商业引擎更适用于较大流量的网站。鉴于此，开源搜索引擎也许可以对中、小流量的网站提供一个替代方案，尤其是，因为它们提供了这些网站所需要的大部分功能。此外，它们还带来开源运动所带来的好处，这就是，免费、积极的软件维护，以及对特殊应用可以自定义代码等。

目前，有很多开源搜索引擎可以使用，每种搜索引擎都有不同的特性，决定将哪个引擎安装到网站上时需要考虑这些特性 [1776]。这些搜索引擎可以根据多种方式进行分类，包括使用的编程语言、如何保存索引（倒排索引、数据库或者其他索引结构）、搜索能力（布尔检索、模糊检索、使用词干提取）、排序方法、可以索引的文件种类（HTML、PDF、纯文本），以及在线增量索引的可能性。其他需要考虑的重要因素包括软件的最后更新日期、目前版本，以及负责维护搜索引擎的项目的活跃水平。例如，一个开源搜索引擎最近没有更新，可能会影响为了网站的特殊应用而定制其代码。此外，还要重点考虑搜索引擎在不同负载下的性能，以及随着数据量的增大性能如何下降。例如，我们可能需要分析索引时间与数据量的关系、索引期间所需要的资源量，以及检索阶段的性能。 737

在本附录中，我们提供了据我们所知的、最详尽的开源搜索引擎比较。我们仔细分析了 27 种不同的搜索引擎，并在以下方面对它们进行了比较：索引时间、检索性能，以及这些指标在不同种类查询和不同文档集下如何变化。我们的目标是对哪种开源搜索引擎最适合给定的应用提供决策支持。

在 A.2 节，我们介绍本章中所使用的搜索引擎。在 A.3 节，我们介绍实验所使用的方法。在 A.4.1 节～A.4.4 节，我们给出了不同实验得出的结果。在 A.4.5 节，我们给出了结果的分析。最后 A.5 节进行了总结。

A.2 搜索引擎

目前有一些免费的搜索引擎可以下载和使用，其中许多是开源的。这里，我们仔细分析了以下 27 个搜索引擎：

ASPSeek、BBDBot、Datapark、ebhath、Eureka、HtDig、Indri、ISearch、Lucene、Managing Gigabytes（MG）、MG4J、mnoGoSearch、MPS Information Server、Namazu、Nutch、Omega、OmniFind IBM Yahoo! Ed.、OpenFTS、PLWeb、SWISH-E、SWISH++、Terrier、WAIS/freeWAIS、WebGlimpse、XML Query Engine、Zebra 和 Zettair。表 A-1 列出了它们的相关细节。

表 A-1 我们初步考虑的 27 个开源搜索引擎（2009 年年底给出的最新版本）

搜索引擎	更新	版本	观察
ASPSeek	2002	N/A	项目中止
BBDBot	2002	N/A	最后更新于 2002 年，之后没有任何活动
Datapark	13/03/2006	4.38	
ebhath	N/A	N/A	网站不存在
Eureka	N/A	N/A	网站不工作
HtDig（ht://Dig）	16/06/2004	3.2.0b6	
Indri	22/06/2009	2.10	
ISearch	02/11/2000	1.75	根据网站说明，虽然软件还能下载，但已不再活跃地维护
Lucene	05/11/2009	2.9.1	
Managing Gigabytes（MG）	01/08/1999	1.2.1	
MG4J	06/06/2009	3.0	
mnoGoSearch	29/10/2009	3.3.9	
MPS Inform. Server	01/09/2000	6.0	
Namazu	23/09/2009	2.0.20	
Nutch	23/03/2009	1.0	Lucene 项目的子项目
Omega	08/04/2006	0.9.5	Omega 是一个使用了 Xapian 库的应用
OmniFind IBM Yahoo!	2009	8.4.2	
OpenFTS	05/04/2005	0.39	
PLWeb	16/03/1999	3.0.4	在 2000 年，AOL 搜索发信宣称，代码不再可用
SWISH-E	04/04/2009	2.4.7	
SWISH++	25/01/2008	6.1.5	
Terrier	29/01/2009	2.2.1	
WAIS & freeWAIS	N/A	N/A	软件过期
WebGlimpse	19/12/2008	4.18.6	使用 Glimpse 作为索引器
XML Query Engine	02/04/2005	0.69	这是 XML 搜索引擎
Zebra	05/11/2009	2.0.42	这是 XML 搜索引擎
Zettair	09/2006	0.9.3	

A.2.1 搜索引擎初步选择

根据收集的信息，在开始阶段就可以放弃一些项目。放弃一个搜索引擎的原因包括：项目已经过期（例如，最后更新日期在 2000 年以前），项目不再维护或已经无效，不能从项目中得到信息。由于这些原因，我们在开始阶段放弃了以下引擎：

- ASPSeek
- BBDBot
- ebhath
- Eureka
- ISearch
- MPS Information Server
- PLWeb
- WAIS/freeWAIS

我们还剩下 19 个引擎进行后续研究。

在某些情况下，一个项目被拒绝是因为其他因素。例如，虽然 MG 项目（“Managing Gigabytes”中给出了介绍 [1709]）构成了这个领域中重要的工作之一，它不再被继续考虑

是因为这个引擎自 1999 年后就没有更新。另外一个特殊的例子是 Nutch 项目。Nutch 搜索引擎是基于 Lucene 搜索引擎的，它只是使用了由 Lucene 提供的 API。由于这个原因，我们只需要对 Lucene 项目进行分析。此外，放弃 XMLQuery Engine 和 Zebra，是因为它们重点在结构化数据（XML），而不是半结构化数据，如 HTML。因此，我们进一步放弃了以下的引擎：

- Managing Gigabytes（MG）
- Nutch
- XML Query Engine
- Zebra

剩下 15 个引擎进行后续研究。 738

关于索引时间的初步试验表明，有些引擎花费了 3～6 倍时间来索引标准参考集，即使这些集合很小。因此，我们不需要在更大规模的性能实验中考虑它们。在这一阶段去除的引擎是：

- Datapark
- mnoGoSearch
- Namazu
- OpenFTS
- Glimpse

还剩下 10 个引擎进行后续研究。

也就是说，从最初的 27 个搜索引擎，我们根据一组初步选择标准排除了 17 个。这些标准代表了开源搜索引擎，作为一个可行性选择需要满足的重要需求。因此，我们留下了以下 10 个搜索引擎进行进一步研究： 739

- HtDig
- Indri
- Lucene
- MG4J
- Omega
- OmniFind
- SWISH-E
- SWISH++
- Terrier
- Zettair

接下来，对这 10 个引擎进行简单的描述，并说明我们在比较时所使用的版本：

- **HtDig**——一系列用来索引和搜索网站的工具［783］。它提供了命令行工具和 CGI 界面来进行搜索。虽然有新的版本，但是根据网站的介绍，我们使用的版本 3.1.6 是最快的一个。
- **Indri**——使用 Lemur［1003］项目构建的搜索引擎，它是为语言模型和信息检索研究［809］而设计的工具。这个项目是由马萨诸塞大学和卡内基梅隆大学联合开发的（版本 2.4）。
- **Lucene**——文本搜索引擎工具库，作为 Apache 软件基金会产品的一部分［1061］。因为它是一个工具库，其他一些应用也使用它，例如 Nutch 项目［1217］。在我们的工作中，使用该工具库所捆绑的简单应用来索引文档集（版本 1.9.1）。

- **MG4J**——（Managing Gigabytes for Java）是一个面向大规模文档集的全文索引器，由意大利米兰大学开发［1074］。作为副产品，它们提供处理字符串、位级别I/O等优化的通用类库（版本1.0.1）。
- **Omega**——架构在Xapian［1725］上的一个应用，它是一个开源概率信息检索工具库。Xapian使用C++编写，但是可以在不同的语言上调用（Perl、Python、PHP、Java、TCL、C#）（版本0.9.5）。
- **IBM Omnifind Yahoo! Edition**——可以快速部署的企业网搜索软件［804］。它结合了基
740 于Lucene搜索引擎的企业网搜索和基于Yahoo!搜索引擎的Web搜索（版本8.4.0）。
- **SWISH-E**——（Simple Web Indexing System for Humans - Enhanced）是一个开源索引和检索引擎［1550］。它是SWISH的扩展版本，由Kevin Hughes编写（版本2.4.3）。
- **SWISH++**——基于Swish-E的索引和检索工具，但是完全由C++重写。它具有Swish-E的绝大部分特征［1549］，但不是全部（版本6.1.4）。
- **Terrier**——（TERabyte RetrIEveR）是一个模块化平台，提供了面向Web、企业网和桌面搜索引擎的快速开发，由苏格兰格拉斯哥大学开发［1574］。它包含了索引、查询和评价标准TREC文档集等功能（版本1.0.2）。
- **Zettair**——（之前叫做Lucy）由RMIT大学搜索引擎组开发的文本检索引擎，它可以处理大规模文本（版本0.9.3）。

A.2.2 特征

我们选择的每个搜索引擎都可以用它们实现的特征和它们在不同场景下的性能来刻画。我们选择了以下13个通用特征用来描述每个搜索引擎：

- **存储（Storage）**——表示索引器采用的存储索引的方法，可以使用数据库引擎或者简单的文件结构（例如，倒排索引）。
- **增量索引（Incremental Index）**——表示索引器是否具有在已有索引上增加文件，而不需要重新生成整个索引的能力。
- **结果摘录（Results Excerpt）**——表示引擎是否可以利用结果生成摘录（片段）。
- **结果模板（Results Template）**——有些引擎可以使用模板对查询结果进行分析。
- **禁用词（Stop words）**——表示索引器是否可以使用禁用词列表来排除某些出现频率过高的项。
- **文件类型（Filetype）**——索引器可以分析的文件类型。引擎可以分析的标准文件格式应该是HTML。
- **词干提取（Stemming）**——表示索引器/检索器是否具有单词的词干提取能力。
- **模糊检索（Fuzzy Search）**——表示引擎是否可以用模糊的方式处理查询，即不需要与查询完全匹配。
- **排列（Sort）**——将结果按照多种标准进行排列。
- **排序（Ranking）**——表示引擎是否可以基于排序函数产生结果。
- **检索类型（Search Type）**——搜索引擎可以处理的检索类型，以及是否可以接受查
741 询操作符。
- **索引器语言（Indexer Language）**——索引器所使用的编程语言。这个信息对扩展引擎所提供的功能或者将它整合到已有平台是有用的。
- **版权（License）**——决定使用和修改索引器和搜索引擎的条件。

对于这10个在我们最终列表中的引擎，我们检查了它们符合上面所列出的哪些特征。

表 A-2 总结了这些信息，给出了 10 个引擎的初步比较。为了支持从它们中进一步选择，需要用性能比较的结果来补充这些信息，接下来，我们将进行讨论。

表 A-2 最终列表中的 10 个开源搜索引擎的主要特征

搜索引擎	存储 (f)	增量索引	结果摘录	结果模板	禁用词	文件类型 (e)	词干提取	模糊检索	排列 (d)	排序	搜索类型 (c)	索引器语言 (b)	版权 (a)
HtDig	1	■	■	■	■	1，2	■	■	1	■	2	1，2	4
Indri	1	■	■	■	■	1，2，3，4	■	■	1，2	■	1，2，3	2	3
Lucene	1	■	□	□	■	1，2，4	■	■	1	■	1，2，3	3	1
MG4J	1	■	■	■	■	1，2	■	□	1	■	1，2，3	3	6
Omega	1	■	□	■	■	1，2，4，5	■	□	1	■	1，2，3	2	4
OmniFind	1	■	■	■	■	1，2，3，4，5	■	■	1	■	1，2，3	3	5
SWISH-E	1	■	□	□	■	1，2，3	■	■	1，2	■	1，2，3	1	4
SWISH++	1	■	□	□	■	1，2	■	□	1	■	1，2，3	2	4
Terrier	1	□	□	□	■	1，2，3，4，5	■	■	1	■	1，2，3	3	7
Zettair	1	■	■	□	■	1，2	■	□	1	■	1，2，3	1	2

(a) 1：Apache，2：BSD，3：CMU，4：GPL，5：IBM，6：LGPL，7：MPL，8：Comm，9：Free
(b) 1：C，2：C++，3：Java，4：Perl，5：PHP，6：Tcl
(c) 1：短语，2：布尔，3：通配符
(d) 1：排序，2：日期，3：无
(e) 1：HTML，2：plain text，3：XML，4：PDF，5：PS ■可用
(f) 1：文件，2：数据库 □不可用

A.2.3 评价

正如之前所讨论的，每个搜索引擎都有多个与其他搜索引擎不同的特性。为了以不同的方式对搜索引擎进行比较，我们定义了一个评价过程，目的是给每个搜索引擎一个客观的等级得分（grade score）。然而，最终的选择依赖于应用的特殊需求。例如，可以从可用性角度评价，即采用黑盒方式使用这个引擎的简单程度，对引擎进行定制的简易程度。举例来说，Lucene 的目的是提供索引和搜索的 API，但是，如果你需要将 Lucene 作为一个前端系统，那么你必须关注子项目 Nutch。另一种可能是分析这些引擎共同的特性，例如索引和检索性能，因为这些特征更容易分析。然而，必须仔细考虑它们，因为它们不是唯一可用的特性。 742

我们选择对一些可以量化比较的参数进行说明，例如，索引时间、索引大小、资源消耗、查询处理时间，和精度-召回率图。最后，我们介绍多个用例和每个引擎可能的替代方案。

A.3 方法

这个研究的主要目的就是比较开源搜索引擎在不同场景下的性能（例如，使用不同的文档集大小），利用通常的标准对它们进行评价。为了进行这个基准测试，我们的研究分为以下步骤：

1）得到由 HTML 表示的文档集。
2）确定一个用来监测搜索引擎性能的工具。
3）安装和配置每个搜索引擎。
4）索引每个文档集。
5）处理和分析索引结果。
6）执行一组预先选择的检索任务。
7）处理和分析检索结果。

A.3.1 文档集

为了比较不同的搜索引擎，我们考虑不同大小的文档集，从不到1GB的文本到10GB以上的文本。另一个要求是使用的文件类型。在这方面，常见的要求支持的文件类型是HTML。为了产生大规模的HTML文档集，可以使用爬虫。但是，因为我们的目的是搜索引擎的索引和排序能力，所以我们决定使用本地文档集。

第一个本地文档集是TREC-4，它包含从《The Wall Street Journal》（华尔街日报）、《Associated Press》（美联社）、《Los Angeles Times》（洛杉矶时报）和其他出版物中抽取的一些文件。每个文件由SGML格式构成，所以需要分析它们来产生HTML文档集。接下来，用这个文档集产生了以下三个子集：

- 第一个子集包含1549篇文档，占用750MB（兆字节）。
- 第二个子集包含3193篇文档，占用1.6GB（吉字节）。
- 第三个子集包含5572篇文档，占用2.7GB。

743 我们把它们称为“三个TREC-4子集”（three TREC-4 subcollections）。

为了进一步扩大比较的范围，我们还考虑了更大规模的文档集，WT10g TREC Web（WebTREC）语料库。它包含1 692 096篇文档，存放在5117个文件中，总计占用10.2GB。这个文档集合使用叠加方式，产生以下4个子集：

- 第一个子集占用2.4GB。
- 第二个子集占用4.8GB。
- 第三个子集占用7.2GB。
- 第四个子集占用10.2GB。

我们把它们称为“四个WT10g子集”（four WT10g subcollections）。

A.3.2 评价测试

我们在选择的文档集上执行了四个不同的评价测试。这些测试是：

- **Test A（测试A）-索引**。第一个测试包括每个搜索引擎对文档集构建索引，记录花费的时间和资源消耗，即内存和CPU。
- **Test B（测试B）-增量索引**。第二个测试包括比较构建增量索引的时间需求。
- **Test C（测试C）-搜索性能**。第三个测试包括比较引擎的查询处理时间，并分析它们在每个文档集上的性能。
- **TestD（测试D）-搜索质量**。第四个测试包括分析每个引擎产生的结果质量，使用精度-召回率指标。

最后，我们根据所有引擎完成上述测试的情况来比较它们。

A.3.3 实验设置

我们进行测试所使用的计算机的主要参数为：

- Pentium 4HT 3.2 GHz处理器。
- 2.0GB内存。
- SATA硬盘。
- 744 Debian Linux（Kernel 2.6.15）。

为了分析每种引擎在处理索引时的资源消耗，需要使用监控工具。有一些开源的监控工具可以使用，例如“Load Monitor”[1045]和“QOS”[1310]。然而，对我们这个工作，简单的监控工具已经足够。所以，我们实现了一个简单的后台程序来记录给定进程的 CPU 和内存的消耗情况。之后，对收集的这些信息进行分析，生成可以用 Gnuplot 显示的数据。

A.4 实验结果

A.4.1 Test A-索引

我们首先考虑索引三个 TREC-4 子集的问题。它们都比较小，为 10 个引擎中哪些具有好的索引能力，哪些没有提供了初步了解。接下来，我们考虑索引四个 WT10g 子集的问题。

1. 索引三个 TREC-4 子集

在本阶段的测试包含使用所有搜索引擎索引三个 TREC-4 子集，记录花费时间和资源消耗（CPU、内存和硬盘上索引大小）。在每个阶段之后，分析由此产生的时间，排除不能在合理时间完成索引的搜索引擎。根据比较分析的结果，我们非严格地定义了“具有合理（reasonable）索引时间的索引器”。

（1）索引时间

图 A-1 给出了在三个 TREC-4 子集上的索引时间。对于 750MB 文档集，搜索引擎的索引时间在 1～32 分钟。对于 1.6GB 文档集，它们的索引时间从 2 分钟～1 小时。对于 2.7GB 文档集，它们的索引时间从 5 分钟～1 小时，但 OmniFind 和 Omega 引擎出现异常。OmniFind 的索引时间超过 2 小时，Omega 的索引时间超过了 17 小时！

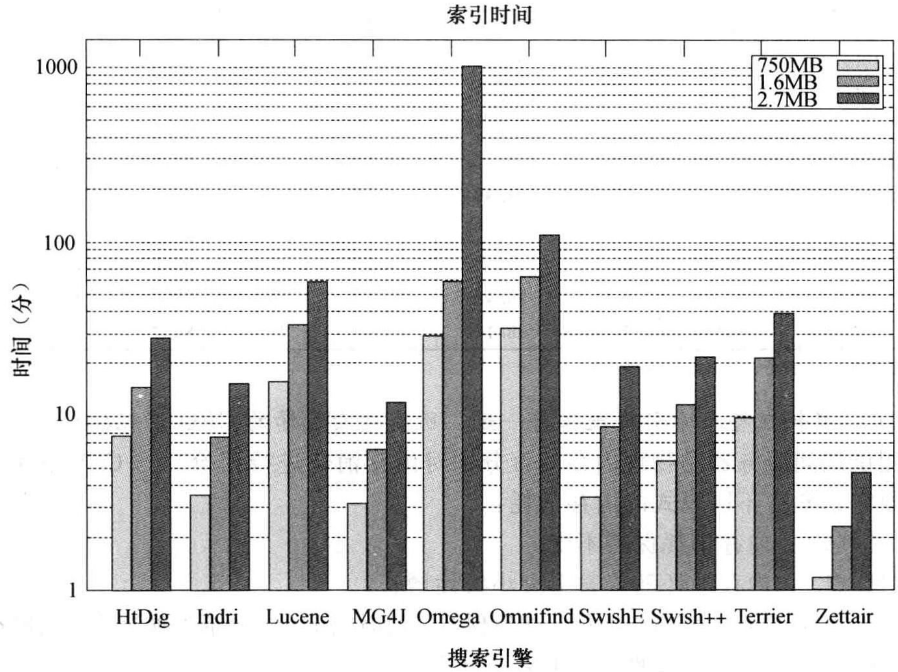


图 A-1 Test A-索引。索引三个 TREC-4 子集的时间

也就是说，10 个引擎中，8 个引擎对 TREC-4 子集索引任务时，有合理的索引时间：

- HtDig
- Indri
- Lucene
- MG4J
- Swish-E
- Swish++
- Terrier
- Zettair

我们进一步使用更大的 WT10g 子集来评价它们的索引能力。

我们还需要提到的是，使用 TREC-4 子集的索引测试排除了下列引擎：Datapark、Glimpse、mnoGoSearch、Namazu 和 OpenFTS。对于较小的 750MB 文档集，它们的索引时间从 77 分钟到大于 320 分钟。也就是说，相比于其他引擎，它们的索引性能更差。一个重
745 要的发现是，所有用数据库来存放索引的搜索引擎的索引时间比其他的要长。

(2) 内存和 CPU 消耗

我们使用 A.3 节中提到的简单监测工具，分析每个搜索引擎在索引阶段的性能，考虑内存和 CPU 消耗。表 A-3 中给出了结果。对于 CPU 消耗，得到了如下结论。

表 A-3 Test A-索引。最大 CPU 和内存（RAM）消耗，RAM 模式（内存消耗模式），以及三个 TREC-4 子集的索引大小

索引大小	750MB			1.6MB			2.7MB		
搜索引擎	最大 CPU 消耗	最大 RAM 消耗	内存使用	最大 CPU 消耗	最大 RAM 消耗	内存使用	最大 CPU 消耗	最大 RAM 消耗	内存使用
HtDig	100.0%	**6.4%**	C	100.0%	**6.4%**	C	88.9%	**6.4%**	C
Indri	100.0%	7.3%	L-S	97.5%	8.0%	L-S	88.6%	9.7%	L-S
Lucene	99.4%	20.0%	L	100.0%	38.3%	L	99.2%	59.4%	L
MG4J	100.0%	23.4%	C	100.0%	48.0%	C	100.0%	70.4%	C
Omega	100.0%	26.8%	L	99.2%	52.1%	L	94.0%	83.5%	L-C
OmniFind	78.4%	17.6%	S	**83.3%**	18.3%	S	83.8%	19.5%	S
Swish-E	100.0%	16.2%	L	98.9%	31.9%	L	98.8%	56.7%	L
Swish++	99.6%	24.8%	S	98.5%	34.3%	S	98.6%	54.3%	S
Terrier	99.5%	58.1%	S-C	99.4%	78.1%	S-C	98.7%	86.5%	S-C
Zettair	**77.2%**	20.2%	L	98.1%	22.3%	L	**82.7%**	23.1%	L

内存消耗模式：C 不变，L 线性，S 步进。

CPU 消耗在整个索引阶段保持一致，所有搜索引擎的 CPU 消耗接近 100%。

采用测试期间服务器总物理内存的百分比对内存消耗进行评价。与 CPU 消耗的情况不同，我们观察到 6 种不同模式的内存消耗：

- 不变（C）：内存消耗保持不变。
- 线性（L）：内存消耗随着索引大小线性增长。
- 步进（S）：内存消耗最初增长，然后保持一段时间不变，之后按照这个模式继续增长。
- 线性-步进（L-S）：线性增长和步进增长相结合。
- 线性-不变（L-C）：线性增长和不变相结合。

746 - 步进-不变（S-C）：步进增长和不变相结合。

对于内存消耗，我们的结论是：

- HtDig 和 MG4J 在整个过程中稳定地使用内存（C 模式）。
- Lucene、Omega、Swish-E 和 Zettair 的内存消耗是线性增长（L 模式）。
- Swish++和 OmniFind 呈现了步进方式（S 模式）。
- Indri 早期内存消耗是线性增长，之后突然降低了内存消耗量，之后重新开始以线性方式消耗内存（L-S 模式）。
- Terrier 呈现的模式是起初步进方式增长，之后突然下降，接下来保持内存消耗不变一直到索引结束（S-C 模式）。
- Omega 在索引 2.7GB 子集的开始时是线性增长模式，之后在达到一定数量后，使用量保持不变（L-C 模式）。

（3）索引大小

表 A-4 给出了 10 个搜索引擎的索引大小。对于索引大小，我们得到如下结论：

- Lucene、MG4J、Swish-E、Swish++和 Zettair 生成的索引大小是文档集大小的 25%～35%。
- Indri 和 Terrier 生成的索引是文档集大小的 50%～55%。
- HtDig、Omega 和 OmniFind 生成的索引是文档集大小的 100%以上。 747

表 A-4 Test A-索引。对于三个 TREC-4 子集所生成的索引的相对大小

搜索引擎	索引大小			搜索引擎	索引大小		
	750MB	1.6GB	2.7GB		750MB	1.6GB	2.7GB
HtDig	108%	92%	104%	OmniFind	175%	159%	171%
Indri	61%	58%	63%	Swish-E	31%	28%	31%
Lucene	**25%**	**23%**	**26%**	Swish++	30%	26%	29%
MG4J	30%	27%	30%	Terrier	51%	47%	52%
Omega	104%	95%	103%	Zettair	34%	31%	33%

2. 索引四个 WT10g 子集

我们使用在 TREC-4 子集中有合理索引时间的 8 个引擎，试图索引整个 WT10g 文档集（10.2GB）。这个集合包含一组文件，每个文件中由一组包含实际 HTML 页面的记录组成。对于整个 WT10g 文档集的索引，我们得到如下的观察结果：

- 只有 Indri、MG4J、Terrier 和 Zettair 有能力直接索引 WT10g 文件，不需要进行干预。此外，它们也是仅有的可以在线性时间内完成任务的引擎。其他的搜索引擎不能扩展到这个程度，或者由于内存缺乏而崩溃。
- HtDig 和 Lucene 需要首先对文档中的文件进行分析，将 HTML 文档抽取出来。此外，它们使用的时间比预想的多 7 倍，比最快的引擎（Zettair）慢了近 20 倍。
- Swish-E 和 Swish++也需要将文档集中的文件进行预处理，由于“缺乏内存”的错误使得程序崩溃。

基于这些结果，我们分析了在四个 WT10g 子集上，Indri、MG4J、Terrier 和 Zettair 所用的索引时间。图 A-2 给出了结果。我们看到四个引擎的索引时间随着文档集大小的增长而线性增长，这很好。 748

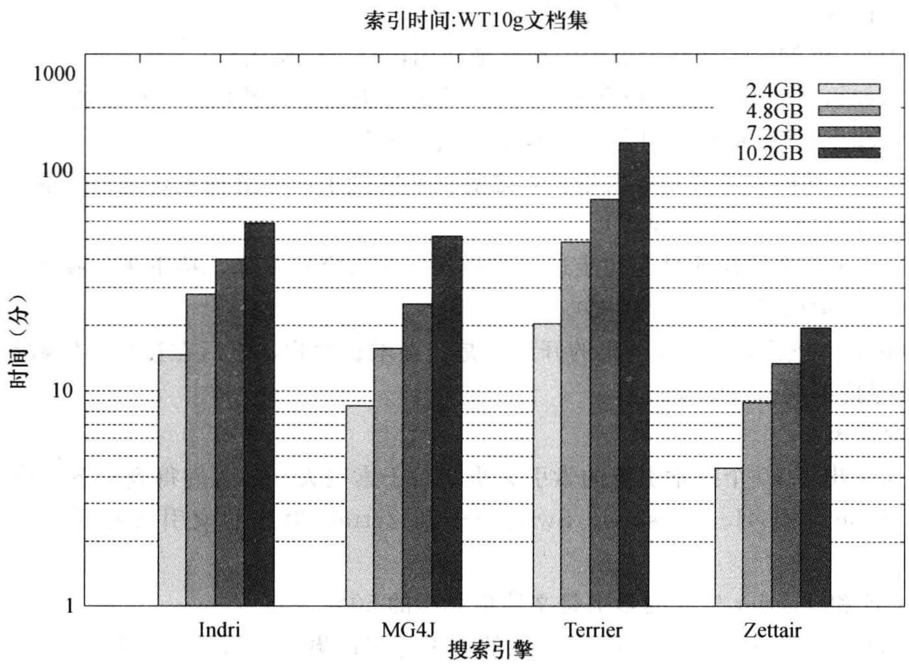


图 A-2 Test A-索引。Indri、MG4J、Terrier 和 Zettair 索引 4 个 WT10g 子集所用的时间

A. 4. 2 Test B-增量索引

我们还在各种大小的情况下比较了增量索引所用的时间：初始文档集大小的 1%、5% 和 10%。增量索引在 1.6GB 文档集中产生。对于这些测试，我们比较了 HtDig、Indri、Swish-E 和 Swish++。图 A-3 给出了增量索引时间的比较图。我们注意到所有四个引擎都可以高效地处理增量索引。

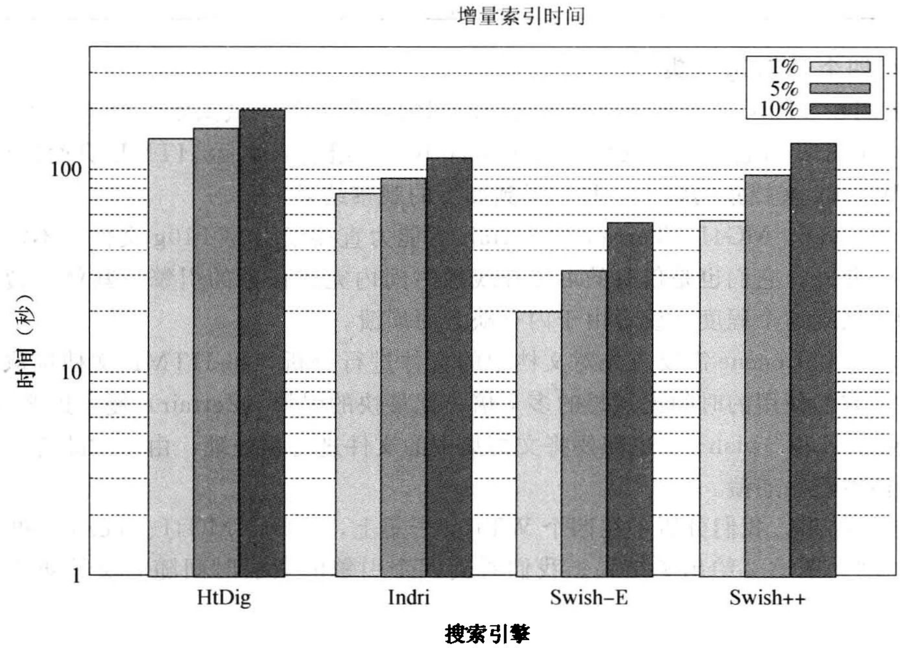


图 A-3 Test B-增量索引。4 个搜索引擎用来处理增量索引的时间

A.4.3 Test C-搜索性能

搜索性能和搜索引擎处理查询的速度以及它需要的资源相关。为了测试这一点，我们使用了两组根据文档中所包含的单词随机生成的查询。第一组包含 100 个单个词查询。第二组包含了 100 个两个词查询。实验在三个 TREC-4 子集上进行，使用具有合理索引时间的 8 个搜索引擎：HtDig、Indri、Lucene、MG4J、Swish-E、Swish++、Terrier 和 Zettair。我们记录了每个引擎的平均查询处理时间和检索比率，定义为：

$$检索比率=\frac{引擎检索的文档个数}{所有引擎检索的文档总数}$$

为了生成查询，我们从每个文档集的词汇表中随机选择了 1 个或者 2 个单词（不包含禁用词），使用多种单词分布：

- 原始的单词分布（幂率）。
- 5%高频词上的均匀分布。
- 30%低频词上的均匀分布。 749

1. 查询处理时间和检索比率

在提交了这些单个词和两个词的查询组（每组包含 100 个查询）后，对每个文档集，我们计算平均查询处理时间和相应的检索比率。对于两个词的查询，我们考虑匹配任何一个单词（使用 OR 操作符）。图 A-4 中给出了这 8 个引擎在 2.7GB 文档集上的平均响应时间图。平均说来，查询处理时间对于单个词和两个词的查询差别在 1.5～2 倍，是线性关系。最快的搜索引擎是 Indri 和 Lucene，其次是 MG4J 和 Zettair。所有系统的检索比率都比较接近，但是，当文档集变大时，检索比率迅速下降。

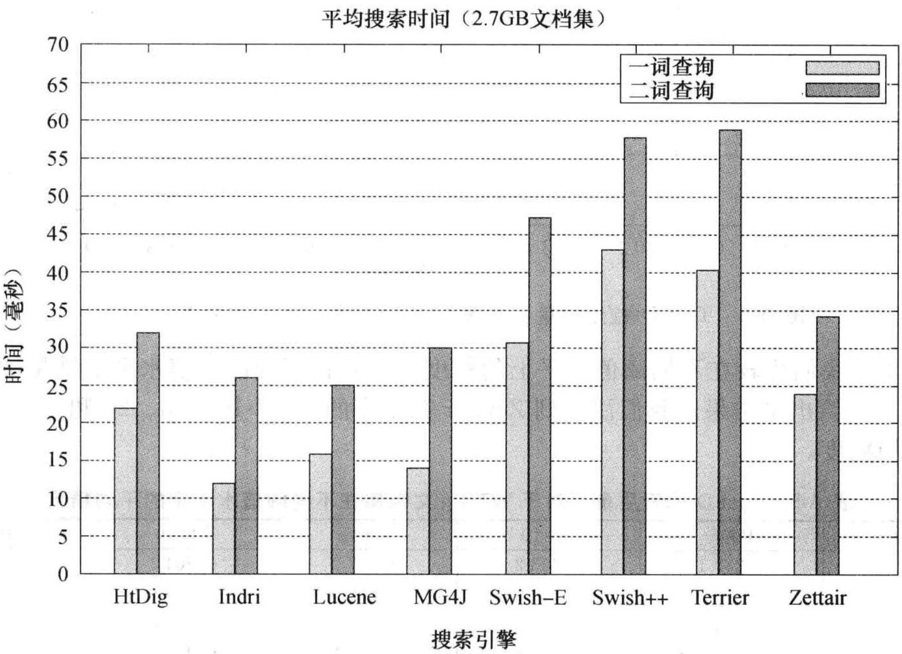


图 A-4 Test C-搜索性能。8 个搜索引擎的平均查询处理时间（2.7GB 文档集）

2. 内存（RAM）消耗

在搜索阶段，我们观察到 4 种不同的行为模式。Indri、Lucene、MG4J、Terrier 和

Zettair 使用不变的内存（1%～2%的内存），与检索文档集的大小无关。Swish++的内存使用是线性增长，对于每个文档集分别到 2.5%、3.5%和 4.5%。Swish-E 和 HtDig 使用了更
750 多的，但是不变的内存。Swish-E 使用了 10.5%的内存，HtDig 使用了 14.4%的内存。

A.4.4 Test D-搜索质量

为了评价不同引擎给出的结果质量，我们使用 WT10g 文档集。为此，我们使用了 50 个 TREC-2001Web Track "主题相关性任务"中的主题（使用只包含标题的查询），以及对应的相关性评价。我们没有使用词干提取，也没有去除禁用词，并且假设查询项间包含 OR 操作。结果的处理使用 trec_eval 软件，它提供标准 NIST 评价，并且可以免费获得。

我们关注在 11 点标准召回率上插值后的平均精度值（参见 4.3.1 节）。对于这个测试，我们选择可以在完整的 WT10g 文档集上生成索引的 4 个引擎进行，它们是：Indri、MG4J、Terrier 和 Zettair。图 A-5 给出了结果。我们注意到，在搜索性能测试上很高效的 Indri 和 Zettair，也给出了最好的结果。

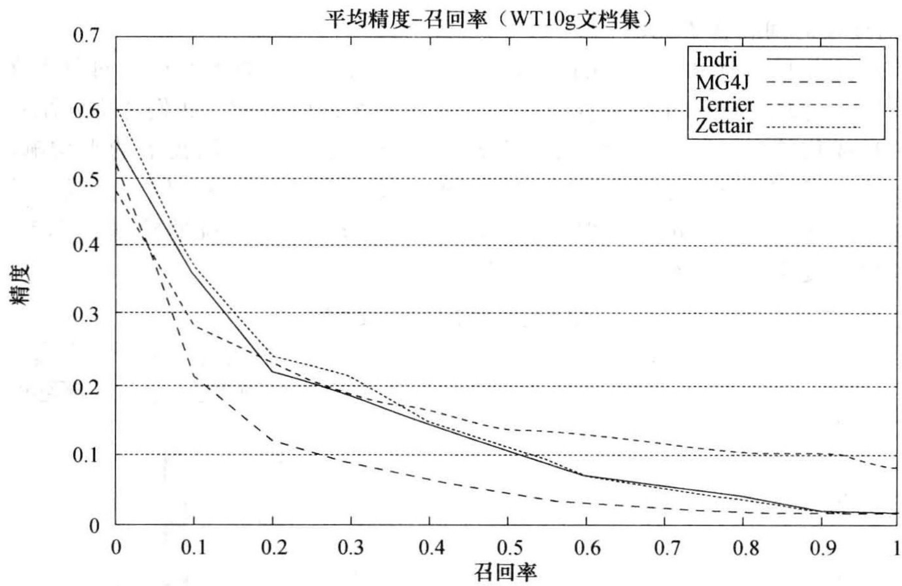


图 A-5 Test D-检索质量。在 WT10g 文档集上的平均精度-召回率

接下来，我们比较在不同阈值水平上的精度，看看在不同的阈值水平上结果的质量如何变化。表 A-5 给出了结果。我们注意到 Zettair 在排序的前 5、10、15、20 和 30 位置得到了
751 最好的平均精度。

表 A-5 Test D-检索质量。对于 WT10g 文档集在不同阈值水平上的平均精度

搜索引擎	P@5	P@10	P@15	P@20	P@30
Indri	0.285 1	0.253 2	0.217 0	0.201 1	0.180 1
MG4J	0.248 0	0.210 0	0.180 0	0.160 0	0.134 0
Terrier	0.280 0	0.240 0	0.213 0	0.210 0	0.193 0
Zettair	**0.324 0**	**0.268 0**	**0.250 7**	**0.231 0**	**0.199 3**

A.4.5 全局评价

通过我们的实验，对选取的 10 个开源搜索引擎的性能进行了更深入的评价（相比初始

的 27 个引擎集合)，得到了一系列的对比结论。在本节中，我们对这些结果进行总结。关于索引，我们的主要结论是：

- HtDig、Indri、Lucene、MG4J、Swish-E、Swish++、Terrier 和 Zettair 比其他引擎索引速度快。
- Lucene、MG4J、Swish-E、Swish++和 Zettair 产生的索引占文档集大小的 25%～35%。
- Terrier 产生的索引大小是文档集的 50%～55%。
- HtDig、Omega 和 OminiFind 产生的索引大小超过文档集本身的大小。 752

关于索引阶段的内存消耗和行为模式，我们的主要结论是：

- HtDig 和 Lucene 的内存使用不变（从 30～120MB)。
- MG4J、Swish-E、Swish++和 Terrier 使用了更多的内存，且内存使用呈现线性增长；对于最小的文档集达到 320～600MB，对于最大的文档集接近 1GB。

在测试的第二部分，我们发现，对于给定的文档集和查询类型（单一词或者两个词)，搜索引擎有相似的查询处理时间，如下所述：

- 对于单个词的查询，平均处理时间为 10～90ms。
- 对于两个词的查询，平均处理时间为 10～110ms。
- Indri 和 Lucene 的平均查询处理时间最小。

在更大的 WT10g 文档集上进行测试，我们可以得到结论：

- 只有 Indri、MG4J、Terrier 和 Zettair 有能力索引整个 WT10g 文档集，且相对 TREC-4 文档集，索引性能没有大的退化。
- Zettair 有最快的索引器，它的精度-召回率图也与 Indri、MG4J 和 Terrier 近似。

A.5 结论

这个研究展示了比较不同开源搜索引擎的方法。最初，考虑了 27 个开源搜索引擎。根据需求进行初始筛选，将这个列表降低到 10 个搜索引擎。在小文档集上的进一步实验，将列表缩小到 8 个具有合理索引时间的引擎：HtDig、Indri、Lucene、MG4J、Swish-E、Swish++、Terrier 和 Zettair。 753

表 A-6 给出了具有合理索引时间的引擎的比较排序，考虑如下四个属性：索引时间、索引大小、查询处理（检索）时间和检索（结果）质量。对于前三个属性，使用 2.7GB TREC-4 子集。对于最后一个属性，使用 WT10g 文档集。只有四个搜索引擎有能力全部索引它：Indri、MG4J、Terrier 和 Zettair。我们的主要结论如下。

表 A-6 根据索引时间、索引大小、平均查询处理时间（对于 2.7GB 文档集）和 P@5（对于 WT10g 文档集）对搜索引擎排序。括号中的数字表示引擎在每个属性中的相对位置

搜索引擎	索引时间（小时：分：秒）		索引大小（%）		搜索时间（毫秒）		答案质量 P@5	
HtDig	(6)	0:28:30	(8)	104	(4)	32		—
Indri	(3)	0:15:45	(7)	63	(1)	**19**	(2)	0.285 1
Lucene	(8)	1:01:25	(1)	**26**	(2)	21		—
MG4J	(2)	0:12:00	(6)	60	(3)	22	(4)	0.248 0
Swish-E	(4)	0:19:45	(3)	31	(6)	45		—
Swish++	(5)	0:22:15	(2)	29	(8)	51		—
Terrier	(7)	0:40:12	(5)	52	(7)	50	(3)	0.280 0
Zettair	(1)	**0:04:44**	(4)	33	(4)	32	(1)	**0.324 0**

Zettair 是最完备的引擎之一，因为 1）可以快速地处理大量信息，使用比其他引擎明显少的时间；2）平均精度-召回率图与其他引擎具有非常大的可比性（对于 WT10g 文档集）。

尽管如此，决定使用哪个搜索引擎必须要考虑当前应用的需求，而不只是量化的指标。其余要考虑的因素包括编程语言（例如，是否可能更改源码）以及服务器属性（例如，内存大小）。例如，如果要索引的文档集很大，而且它还会变化（即需要频繁地索引），也许可以选择 Zettair、MG4J 或者 Swish++，因为它们索引建立和搜索阶段都很快。Swish-E 也是一个好的选择。另一方面，如果限制条件是硬盘空间，那么 Lucene 是一个好的选择，因为它只需要使用很少的空间和较少的检索时间。它的缺点是对文档集建立索引的时间。最后，如果文档集不频繁改变，又因为所有搜索引擎具有相似的检索时间，那么可以根据应用所处网站使用的编程语言决定，以便最大限度地减少定制时间。对于 Java，可以选择 MG4J、Terrier 或者 Lucene。

754 对于 C/C++，可以选择 Swish-E、Swish++、HtDig 或 Zettair。

作者简介

主要作者简介

Ricardo Baeza-Yates 于 1989 年在加拿大滑铁卢大学获得计算机科学博士学位，就读期间参加了牛津英语词典项目。在此之前，他于 1983 年在智利大学获得计算机科学学士学位，后来还在该校先后获得计算机科学理学硕士学位（1985 年）、电气工程专业技术职称（1985 年）和电子工程工学硕士学位（1986 年）。20 世纪 90 年代，他担任过两届智利计算机科学学会（Chilean Computer Science Society，SCCC）主席。2000 年，他创立了智利互联网搜索引擎 TodoCL. com，迄今仍然活跃。从 2000—2004 年，他是拉美计算机科学系联合会（CLEI）的主席，伊比利亚-美洲科技合作项目（CYTED）应用电子和信息技术领域的国际协调员。2002—2005 年，他创办并领导了智利大学工学院计算机科学系 Web 研究中心。2004 年年底，他成为西班牙巴塞罗那庞培法布拉大学信息和通信技术系 ICREA 研究教授。自 2006 年以来，他一直担任雅虎欧洲和拉丁美洲研究院的副总裁，领导西班牙巴塞罗那和智利圣地亚哥的实验室，2008 年，他兼管以色列海法的新研究实验室。

他的研究兴趣包括 Web 检索和数据挖掘、索引和搜索算法。他一直是 IEEE 计算机分会管理委员会和 ACM 出版委员会委员。他曾经担任多个主流学术会议的程序委员会主席或联合主席，包括 ACM SIGIR 2002、ACM CIKM 2007、ACM KDD 2009 和 ACM/IEEE/WIC WI/IAT 2009。他也是 ACM SIGIR 2005 和 ACM WSDM 2009 的大会主席或联合主席。他是《ACM TOIS》、《Information Systems》和《Information Processing & Management》等学术期刊的副主编。他一共发表了 250 多篇论著，其中包括多本合著的书籍，例如 1991 年由 Addison-Wesley 出版的《Handbook of Algorithms and Data Structures》第二版，1992 年担任由 Prentice-Hall 出版的《Information Retrieval: Algorithms and Data Structures》一书的联合编辑。他曾获得美洲国家组织向从事精密科学的青年研究人员颁发的奖项（1993 年），并于 1997 年和两位巴西同事因巴西计算机科学最佳研究论文而共同获得 COMPAQ 奖。2003 年，他成为第一位当选智利科学院院士的计算机科学家。2007 年，作为滑铁卢大学的杰出校友，他被授予格雷厄姆奖章，以奖励他在计算领域的创新贡献。2009 年，他被 CLEI 授予拉美地区计算机科学杰出贡献奖并成为 ACM 院士。他是 ACM 院士，IEEE 高级会员，AMS、EATCS、SCCC 和 SIAM 等学会的会员。 755

Berthier Ribeiro-Neto 于 1995 年在加州大学洛杉矶分校获得计算机科学博士学位。在此之前，他在巴西贝洛奥里藏特市的米纳斯吉拉斯联邦大学（UFMG）先后获得数学学士学位、电气工程学士学位和计算机科学硕士学位。1996 年，他到 UFMG 计算机科学系工作，目前是副教授。

2000 年，Ribeiro-Neto 在贝洛奥里藏特共同创办了 Akwan 信息技术公司，这是一家为巴西互联网提供搜索引擎服务的创业公司。2001 年，他从 UFMG 无薪休假，并成为 Akwan 的首席执行官。公司向企业市场销售定制的搜索解决方案，取得蓬勃发展。2005 年，Akwan被谷歌收购，成为谷歌的拉美工程办公室，他目前担任工程总监和现场主管。

Ribeiro-Neto 的主要兴趣是信息检索系统，Web 搜索和社交网络。他曾参与科技部

(MCT) 和国家科学技术发展委员会 (CNPq) 等巴西国家研究机构资助的许多研究项目。他是1998年字符串处理和信息检索研讨会 (String Processing and Information Retrieval Symposium, SPIRE) 和1999年巴西数据库讨论会 (Brazilian Symposium on Databases, SBBD) 的程序委员会主席，以及2009年ACM Web搜索和数据挖掘会议 (ACM Conference onWeb Search and Data Mining, ACM WSDM) 的程序委员会联合主席。他已经在各种会议和期刊上发表了70多篇论文，是ACM信息系统汇刊 (ACM Transactions on Information Systems, ACM TOIS) 的副主编和ACM会员。

撰稿人简介

Eric Brown 在美国佛蒙特大学获得计算机科学学士学位，在马塞诸萨大学获得计算机科学硕士和博士学位。Eric在马塞诸萨大学师从Bruce Croft，是智能信息检索中心的成员。1995年，Eric作为研究人员加入IBM的T.J.Watson研究实验室。Eric在IBM从事信息检索、文档分类、文本分析、问题回答、生物信息学和自动语音识别应用的研究。自2007年起，Eric参与了IBM的DeepQA项目，运用开放域自动问题回答技术，开发Watson问答系统。Watson的目标是实现人类水平的问题回答性能。Eric在许多会议和期刊发表了论文，
756 并持有文本分析和问题回答领域的多项专利。

Carlos Castillo 是在雅虎巴塞罗那研究院的研究科学家。在此之前，他是罗马智慧大学和巴塞罗那庞培法布拉大学的博士后研究人员。2004年，他从智利大学获得博士学位，主要研究方向是网络爬取、Web刻画和Web排序。Castillo博士目前活跃在Web用途和链接挖掘领域。他已在权威刊物和国际会议发表了数篇论文，组织了数次敌对Web信息检索的研讨会和竞赛，并在信息检索领域的主流会议 (WWW、WSDM、SIGIR和CIKM等) 担任程序委员会委员。

Marcos André Gonçalves 是米纳斯吉拉斯联邦大学 (UFMG) 计算机科学系助理教授。他于2004年在弗吉尼亚理工大学获得计算机科学博士学位，1997年在巴西坎皮纳斯州立大学 (UNICAMP) 获得计算机科学硕士学位，1995年在巴西塞阿拉联邦大学 (UFC) 获得计算机科学学士学位。他曾在多本期刊 (《TOIS》、《TIDE》、《IP&M》、《Information Retrieval》和《Information Systems》等) 和会议 (SIGIR、CIKM和JCDL等) 担任审稿人。他的研究兴趣包括信息检索、数字图书馆、通用文本分类和文本挖掘，并在这些领域发表了多篇论文。Marcos是巴西科学院的附属会员。

David Hawking 是澳大利亚堪培拉Funnelback公司的首席科学家。Hawking博士及其团队先后在澳大利亚国立大学 (ANU) 和澳大利亚联邦科学与工业研究组织 (CSIRO) 从事研究工作，后来独立为Funnelback公司，从事企业和Web搜索的商业化研究。David是澳大利亚国立大学的兼职教授和博导。1997—2004年，他和Nick Craswell一起担任TREC Web搜索任务的协调员，全球120多个研究机构使用了他创建并分发的信息检索基准集。他是2003年和2006年ACM SIGIR会议的程序委员会主席。他在澳大利亚国立大学获得博士学位，并被纳沙泰尔大学授予荣誉博士学位。2004年，他获得澳大拉西亚计算机科学研究奖。他的研究兴趣包括分布式信息检索、基于距离的排序、个人元搜索、文档注释、高效检索算法、Web搜索、健康信息的自动质量评级、信息检索评价和企业搜索。

Marti Hearst 是加州大学伯克利分校信息学院教授，并在计算机科学部兼职。她的主要研究兴趣为搜索引擎用户界面、信息可视化、自然语言处理和社会媒体的经验分析。她刚刚完成了关于搜索用户界面的第一本书籍。她先后在加州大学伯克利分校获得计算机科学学

士、硕士和博士学位，并于1994—1997年在施乐公司帕洛阿尔托研究中心（Xerox PARC）担任研究人员。Hearst教授曾在NSF的CISE咨询委员会任职，也是CACM的互联网委员会联合主席。她是《American Heritage Dictionary》用法小组成员和Edge.org专家小组成员。她曾任《Computational Linguistics》、《ACM Transactions on Information Systems》和《IEEE Intelligent Systems》的编委，目前任《ACM Transactions on the Web》和《ACM Transactions on Computer-Human Interaction》的编委。

Mounia Lalmas 在英国格拉斯哥大学计算机科学系担任微软研究院/皇家工程院首席研究员。在此之前，1999年，作为讲师加入伦敦大学玛丽女王学院计算机科学系，之后晋升为信息检索教授。1998年，她在多特蒙德大学担任研究科学家。1995—1997年，她在格拉 757
斯哥大学担任讲师，1997—1998年转为研究员，1996年获得博士学位。她获得了特许信息技术专业资格（Chartered IT Professional，CITP），并担任英国计算机学会（FBCS）院士。她曾任ACM SIGIR的信息主管，现已当选为副主席。她是《ACM TOIS》、《IR》（Springer）和《IP&M》（Elsevier）的编委。她的研究主要集中在交互式复杂异质信息库的智能接口开发和评价，并广泛涵盖HTML、XML和MPEG-7等领域。2002—2007年，她与Norbert Fuhr共同领导了INEX XML检索评价（Evaluation Initiative for XML Retrieval)。这是一个由来自世界各地80多个机构参与的大型项目，负责定义XML检索的性质和评价方法。她多次在CIKM、SIGIR和ESSIR等国际会议上做有关XML检索和评价的报告和讲座。她现在主要从事聚合搜索和弥合数字鸿沟的研究，同时也回归到理论信息检索方向，使用量子理论对交互式信息检索建模。2004年和2006年她在SIGIR担任研讨会主席，2009年担任辅导主席。她还先后担任CIKM 2008和WI/IAT 2009的公共关系（共同）主席，CIKM 2010的研讨会主席，2006年欧洲信息检索研究会议（European Conference on Information Retrieval Research，ECIR）的程序委员会主席，2009年WWW会议XML和Web数据分会的副主席，2008年语境中信息交互会议（Information Interaction in Context，IIiX）和2010年欧洲数字图书馆会议（European Conference on Digital Libraries，ECDL）的大会共同主席。

Yoelle Maarek 于2009年6月加入雅虎以色列实验室，并担任高级研究总监。在此之前，她于2006年3月创办了谷歌海法工程研究中心，并担任工程总监，该中心有近40位研究人员和软件工程师。她在谷歌海法的团队推出了“谷歌建议”，这是近年来Web搜索最鲜明的特色之一，具有查询填充功能，适用于大多数语言，自2008年8月以来已部署在google.com，并应用于YouTube、iGoogle和移动搜索等一系列Google服务。海法团队也在其他领域开发特色服务，如搜索广告和YouTube上的互动注释。在此之前，自1989年以来她在IBM研究中心工作，先后担任了一系列技术和管理职务。她首先在美国纽约的T. J. Watson研究中心，然后在IBM以色列海法的研究实验室工作到2006年2月，在此期间她促成了IBM企业搜索服务。她在IBM的最后两个职位是搜索和协作领域的杰出工程师和部门经理。她在1985年毕业于法国巴黎的国立路桥学校，并从巴黎第六大学获得了计算机科学研究生学位（DEA)。1986—1987年她在纽约哥伦比亚大学做访问博士生。1989年她在海法的以色列理工大学获得了计算机科学博士学位。Yoelle的研究兴趣包括信息检索、Web应用和协作技术。在这些领域中，她已发表了50多篇论文和文章。她活跃在科研界，过去10年中，她在WWW系列会议的多个技术分会担任主席或副主席，在大多数ACM SIGIR会议担任高级或普通程序委员会委员。她还在WWW和SIGIR会议主持了多个研讨会和小组讨论。最近，她和Andrei Broder共同主持2008年WWW会议的讨论活动，和Wolf- 758

gang Nejdl一起担任2009年4月在马德里举行的WWW会议的技术程序联合主席。自2009年以来，Yoelle也是海法大学Caesarea-Rotschild研究所董事局成员和以色列理工大学理事会成员。

Christian Middleton 目前是一名软件工程师。此前，他是庞培法布拉大学计算机科学与数字通信专业的博士研究生，师从Ricardo Baeza-Yates。2004年，他在智利大学获得了硕士学位和计算机科学工程师职称。他的主要兴趣领域是Web挖掘和日志分析。在过去几年里，他先后参加了Web图可视化、用户日志分析和搜索引擎评价等项目。

Gonzalo Navarro 于1998年在智利大学获得计算机科学博士学位，目前是该校计算机科学系教授。他也是Millennium细胞动力学和生物技术研究所研究员。他感兴趣的领域包括算法和数据结构、文本搜索、压缩和度量空间搜索。他负责过一些文本搜索和信息检索的研究项目，受到伊比利亚美洲信息检索研究组（RIBIDI）的互联网研究中心和雅虎研究院资助。Navarro教授担任过多个会议的程序委员会（联合）主席，包括2001年和2005年的字符串处理和信息检索会议（SPIRE），并担任ACM SIGIR 2005的海报主席。2008年他参与创建了相似性搜索及应用会议（Similarity Search and Applications，SISAP）。他是拉丁美洲理论信息学会议（LATIN）和SISAP的指导委员会委员，《Information Retrieval journal》和《ACM Journal of Experimental Algorithmics》编委。他是剑桥大学出版社出版的《String Matching》一书的作者之一，著有15章，编辑了6本国际学术会议论文集，在国际期刊发表了约80篇论文，在国际会议发表了约140篇。

Dulce Ponceleón 拥有斯坦福大学计算机科学硕士和博士学位。她曾在苹果电脑公司先进技术组工作，为QuickTime研究信息检索和音视频压缩技术。她为第一个纯软件视频会议系统做出了关键贡献。她目前在IBM Almaden研究中心工作，管理着内容保护技能中心。她的工作包括多媒体内容分析和索引、视频文摘、语音识别应用、存储系统和内容保护。她对ISO MPEG-7的标准化工作做出了贡献，特别是多媒体的描述方案。她是IBM在4C及高级访问内容系统（AACS）的技术代表。4C已发展为可录制和预录制介质（CPRM/CP-PM）的内容保护标准。Ponceleón博士自2004年以来担任4C技术组主席。AACS是管理存储在下一代预录制或可录制光学介质，为个人电脑和消费电子设备用户提供服务的内容保护标准。Ponceleón博士是一所主要的NSF多媒体学校的科学咨询委员会委员，并担任ACM Multimedia、SPIE、SIGIR、IEEE和一些多媒体研讨会的程序委员会委员。她曾主持了ACM MM 2000的多媒体标准研讨会和ACM MM 2001的流式视频小组讨论，在SIGIR 2002、SIGIR 2005和ICASPP 2006做多媒体信息检索讲座。她在音视频压缩、多媒体信息
759 检索、内容保护、人机接口、数值线性代数和非线性规划方面拥有专利，发表了许多论著。

Edie Rasmussen 目前在加拿大温哥华担任不列颠哥伦比亚大学（UBC）图书馆、档案和信息研究学院教授，并已担任院长六年。在加入UBC之前，她是美国匹兹堡大学信息科学学院教授。她曾在加拿大新斯科舍省达尔豪西大学图书馆和信息研究学院和马来西亚吉隆坡Institiut Teknoloji MARA大学图书馆学院任职，并曾在新加坡南洋理工大学、新西兰惠灵顿维多利亚大学、挪威奥斯陆大学访问。Rasmussen博士一直活跃在信息检索和数字图书馆的研究界，曾任ACM SIGIR、ACM DL、ACM/IEE JCDL和ASIS&T会议的主席。她曾担任美国信息科学与技术学会主席、加拿大信息研究委员会主席，以及图书馆和信息科学教育协会院长理事会联合召集人。她目前的研究兴趣包括文本、多媒体数据库和数字图书馆的信息索引和检索。

Malcolm Slaney 是雅虎研究院的首席科学家，研究多媒体数据的各种处理方法。他在普

渡大学获得计算显像博士学位。他和 A. C. Kak 共同撰写了《Principles of Computerized Tomographic Imaging》，并由 IEEE 出版。最近这本书由 SIAM 再版，列入“应用数学经典”系列。他和 Steven Greenberg 共同编辑了《Computational Models of Auditory Function》。Slaney 博士在加入雅虎之前，曾在贝尔实验室、斯伦贝谢公司位于帕洛阿尔托的研究中心、苹果电脑、Interval Research 实验室和 IBM Almaden 研究中心工作。他也是在斯坦福大学音乐与声学计算研究中心（CCRMA）的（咨询）教授，组织和指导听证研讨会。他的研究兴趣包括听觉建模和感知、多媒体分析和综合、压缩域处理、音乐相似性和音频搜索，以及机器学习。在过去的几年中，他领导 Telluride 神经形态研讨会的听觉组。

Nivio Ziviani 于 1982 年从加拿大滑铁卢大学获得计算机科学博士学位。他是巴西米纳斯吉拉斯联邦大学计算机科学系的名誉教授，主持了信息处理实验室。他是巴西科学院成员，曾荣获巴西国家科学功绩勋章。他是两个高科技创业公司的创始人之一，其中矿业科技集团公司在 1999 年出售给了圣保罗报业集团，Akwan 信息技术公司在 2005 年出售给了谷歌公司。他发表了许多关于算法设计和信息检索的论著，后者是他的主要研究领域。2005 年他担任第 28 届 ACM SIGIR 会议（SIGIR）大会联合主席，并于 1993 年和 Ricardo Baeza-Yates 共同创立了国际字符串处理和信息检索会议（International Conference on String Processing and Information Retrieval，SPIRE）。 760

参考文献

[1] I. Aalbersberg. Incremental relevance feedback. In *Proc of the Fifteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 11–22, Denmark, 1992.

[2] H. Abdi. Kendall rank correlation. In N. Salkind, editor, *Encyclopedia of Measurement and Statistics*, Thousand Oaks, CA, 2007. Sage.

[3] H. Abdi. The Kendall rank correlation coefficient. Technical report, Univ. of Texas at Dallas, 2007.

[4] K. Aberer, F. Klemm, M. Rajman, and J. Wu. An architecture for peer-to-peer information retrieval. In *Workshop on Peer-to-Peer Information Retrieval*, Sheffield, UK, July 2004.

[5] S. Abiteboul. Querying semi-structured data. In F. N. Afrati and P. Kolaitis, editors, *Int. Conf. on Database Theory (ICDT)*, number 1186 in LNCS, pages 1–18, Delphi, Greece, 1997. Springer-Verlag.

[6] S. Abiteboul, M. Preda, and G. Cobena. Adaptive on-line page importance computation. In *Proceedings of the twelfth international conference on World Wide Web*, pages 280–290, Budapest, Hungary, 2003. ACM Press.

[7] M. Abolhassani and N. Fuhr. Applying the divergence from randomness approach for content-only search in xml documents. *Lecture Notes in Computer Science*, 2997:409–420, 2004.

[8] M. Abrams, editor. *World Wide Web: Beyond the Basics*. Prentice Hall, 1998.

[9] M. Abrol, N. Latarche, U. Mahadevan, J. Mao, R. Mukherjee, P. Raghavan, M. Tourn, J. Wang, and G. Zhang. Navigating large-scale semi-structured data in business portals. In *Proceedings of the 27th VLDB Conference*, pages 663–666, Roma, Italy, 2001. http://www.vldb.org/conf/2001/P663.pdf.

[10] A. Adams and A. Blandford. Digital libraries' support for the user's 'information journey'. In *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 160–169, Denver, Colorado, 2005.

[11] E. Adar. User 4xxxxx9: Anonymizing query logs. In *Query Log Analysis: Social and Technological Challenges, Workshop in WWW'07*, 2007.

[12] J. Adiego, G. Navarro, and P. de la Fuente. Scm: Structural contexts model for improving compression in semistructured text databases. In *Proc. 10th International Symposium on String Processing and Information Retrieval (SPIRE 2003)*, LNCS 2857, pages 153–167. Springer, 2003. Extended version appeared in *Information Processing and Management* 43(3), May 2007, pp. 769–790.

[13] J. Adiego, G. Navarro, and P. de la Fuente. Lempel-Ziv compression of structured text. In *Proc. 14th IEEE Data Compression Conference (DCC'04)*, pages 112–121, 2004. Extended version appeared in *JASIST* 58(4), 2007, pp. 461–478.

[14] M. Adler and M. Mitzenmacher. Towards compressing Web graphs. In *Data Compression Conference*, pages 203–212, 2001.

[15] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, 2005.

[16] D. Agarwal and S. Merugu. Predictive discrete latent-factor models for large-scale dyadic data. In *KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 26–35, New York, NY, USA, 2007. ACM.

[17] E. Agichtein, E. Brill, and S. Dumais. Improving Web search ranking by incorporating user behavior information. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–26, New York, NY, USA, 2006. ACM Press.

[18] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. Learning user interaction models for predicting Web search result preferences. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–10. ACM Press, 2006.

[19] A. Agogino and J. Ghosh. Increasing pagerank through reinforcement learning. In *Proceedings of Intelligent Engineering Systems Through Artificial Neural Networks*, volume 12, pages 27–32, St. Louis, Missouri, USA, November 2002. ASME Press.

[20] M. Agosti and A. F. Smeaton, editors. *Information retrieval and hypertext.* Kluwer Academic Publishers, Boston/London/Dordrecht, 1996.

[21] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. Bocca, M. Jarke, and C. Zaniolo, editors, *20th Int Conference on Very Large Databases*, pages 487–499. Morgan Kaufmann Publishers, 1994.

[22] A. Aho and M. Corasick. Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18(6):333–340, June 1975.

[23] AIR Workshops: Adversarial Web Retrieval. http://airweb.cse.lehigh.edu/, 2005.

[24] A. Al-Maskari, M. Sanderson, and P. Clough. The relationship between IR effectiveness measures and user satisfaction. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 773–774, 2007.

[25] S. M. Alessi and S. R. Trollip. *Multimedia for learning: methods and development.* Allyn and Bacon, 2001. 580 pages.

[26] S. Ali, M. Consens, G. Kazai, and M. Lalmas. A common basis for the evaluation of structured document retrieval. In *ACM CIKM International Conference on Information and Knowledge Management, Bremen, Germany*, 2008. In Press.

[27] W. Alink, R. Bhoedjang, P. Boncz, and A. de Vries. XIRAF - XML-based indexing and querying for digital forensics. *Digital Investigation*, 3(Supplement-1):50–58, 2006.

[28] Alis Technologies. Web languages hit parade, 1997.

[29] J. Allan. Incremental relevance feedback for information filtering. In *Proc of the 19th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 270–278, Zurich, Switzerland, 1996.

[30] J. Allan. Perspectives on information retrieval and speech. In *Information Retrieval Techniques for Speech Applications, from the workshop "Information Retrieval Techniques for Speech Applications," held as part of the 24th Annual International ACM SIGIR Conference*, pages 1–10, London, UK, 2002. Springer-Verlag.

[31] J. Allan. HARD Track overview in TREC 2004 high accuracy retrieval from documents. *Proceedings of the Thirteenth Text REtrieval Conference (TREC'04)*, 2005.

[32] B. L. Allen. *Information Tasks: Toward a User-Centered Approach to Information Systems.* Academic Press, San Diego, CA, 1996.

[33] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.

[34] O. Alonso and R. Baeza-Yates. A model for visualizing large answers in WWW. In *XVIII Int. Conf. of the Chilean CS Society*, pages 2–7, Antofagasta, Chile, 1998. IEEE CS Press.

[35] O. Alonso, R. Baeza-Yates, and M. Gertz. Effectiveness of temporal snippets. In *WSSP Workshop at the World Wide Web Conference—WWW'09*, 2009.

[36] O. Alonso and S. Mizzaro. Can we get rid of TREC assessors? Using Mechanical Turk for relevance assessment. In *SIGIR Evaluation Workshop*, 2009.

[37] O. Alonso, D. E. Rose, and B. Stewart. Crowdsourcing for relevance evaluation. *SIGIR Forum*, 42(2):9–15, 2008.

[38] G. Amati. *Probability Models for Information Retrieval based on Divergence from Randomness*. PhD thesis, Department of Computing Science University of Glasgow, 2003. http://www.dcs.gla.ac.uk/~gianni/selectedPapers.html.

[39] G. Amati and C. van Rijsbergen. Probabilistic models of information retrieval based on measuring divergence from randomness. *ACM Transaction on Office and Information Systems (TOIS)*, 20(4), 2002.

[40] Amazon. Mechanical Turk, 2009. http://www.mturk.com.

[41] G. M. Amdahl. Validity of the single-processor approach to achieving large scale computing capabilities. In *Proc. AFIPS 1967 Spring Joint Computer Conf.*, volume 30, pages 483–485, Atlantic City, N.J., Apr. 1967.

[42] S. Amer-Yahia, C. Botev, J. Dörre, and J. Shanmugasundaram. Full-Text extensions explained. *IBM Systems Journal*, 45(2):335–352, 2006.

[43] S. Amer-Yahia, C. Botev, and J. Shanmugasundaram. TexQuery: a full-text search extension to XQuery. In *13th international conference on World Wide Web, New York, NY, USA*, pages 583–594, 2004.

[44] S. Amer-Yahia, P. Case, T. Roelleke, J. Shanmugasundaram, and G. Weikum. Report on the DB/IR panel at SIGMOD 2005. *SIGMOD Record*, 34(4):71–74, 2005.

[45] S. Amer-Yahia, S. Cho, and D. Srivastava. Tree Pattern Relaxation. In *Advances in Database Technology - EDBT 2002, 8th International Conference on Extending Database Technology, Prague, Czech Republic*, pages 496–513, 2002.

[46] S. Amer-Yahia, D. Hiemstra, T. Roelleke, D. Srivastava, and G. Weikum. Ranked XML Querying. In S. Amer-Yahia, D. Srivastava, and G. Weikum, editors, *Workshop on Ranked XML Querying*, number 08111 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2008.

[47] S. Amer-Yahia and M. Lalmas. XML search: languages, INEX and scoring. *SIGMOD Record*, 35(4):16–23, 2006.

[48] A. Amir, S. Srinivasan, and A. Efrat. Search the audio, browse the video: A generic paradigm for video collections. *EURASIP Journal on Applied Signal Processing*, 2003(2):209–222, 2003. doi:10.1155/S111086570321012X.

[49] E. Amitay and C. Paris. Automatically summarising Web sites - is there a way around it? In *Proceedings of the 2000 ACM CIKM International Conference on Information and Knowledge Management*, pages 173–179, McLean, VA, USA, November 2000.

[50] C. Anderson. *The Long Tail: Why the Future of Business Is Selling Less of More.* Hyperion, New York, revised edition, 2008.

[51] T. Anderson, A. Hussam, B. Plummer, and N. Jacobs. Pie charts for visualizing query term frequency in search results. *Proceedings of the Fifth International Conference on Asian Digital Library*, pages 440–451, 2002.

[52] K. Andrews. Visualising cyberspace: Information visualization in the Harmony Internet browser. In *Proceedings '95 Information Visualization*, pages 97–104, Atlanta, Oct. 1995.

[53] K. Andrews, C. Gütl, J. Moser, V. Sabol, and W. Lackner. Search Result Visualisation with xFIND. *Proceedings of User Interfaces to Data Intensive Systems (UIDIS 2001)*, pages 50–58, 2001.

[54] V. N. Anh, O. de Kretser, and A. Moffat. Vector-space ranking with effective early termination. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *Proc. 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 35–42, New Orleans, LA,

Sept. 2001. ACM Press, New York.

[55] V. N. Anh and A. Moffat. Impact transformation: Effective and efficient Web retrieval. In M. Beaulieu, R. Baeza-Yates, S. H. Myaeng, and K. Järvelin, editors, *Proc. 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–10, Tampere, Finland, Aug. 2002. ACM Press, New York.

[56] V. N. Anh and A. Moffat. Simplified similarity scoring using term ranks. In *Proc. SIGIR 2005*, pages 226–233, 2005.

[57] V. N. Anh and A. Moffat. Pruned query evaluation using pre-computed impacts. In *SIGIR'06: Proceedings of the 29th International ACM SIGIR conference on Research and Development in Information Retrieval*, Seattle, WA, USA, 2006.

[58] V. N. Anh and A. Moffat. Pruning strategies for mixed-mode querying. In *CIKM'06: Proceedings of the 15th ACM International conference on Information and Knowledge Management*, Arlington, Virginia, USA, 2006.

[59] P. Anick. Using Terminological Feedback for Web Search Refinement: A Log-Based Study. *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'03)*, pages 88–95, 2003.

[60] P. Anick, J. Brennan, R. Flynn, D. Hanssen, B. Alvey, and J. Robbins. A direct manipulation interface for Boolean information retrieval via natural language query. In *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'90)*, pages 135–150, Brussels, Belgium, 1990.

[61] P. Anick and R. Kantamneni. A longitudinal study of real-time search assistance adoption. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'08)*, pages 701–702, New York, NY, USA, 2008. ACM.

[62] ANSI/NISO Standards. Z39.50-information retrieval: Application service definition and protocol specification. Technical report, International Standard Maintenance Agency, Washington, USA, 1995. See http://lcweb.loc.gov/z3950/agency.

[63] A. Apostolico and Z. Galil, editors. *Combinatorial Algorithms on Words*. Springer-Verlag, New York, 1985.

[64] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan. Searching the web. *ACM Transactions on Internet Technology*, 1(1):2–43, 2001.

[65] M. Araújo, G. Navarro, and N. Ziviani. Large text searching allowing errors. In *Proc. WSP'97*, pages 2–20. Carleton University Press, 1997.

[66] Y. Aridor, D. Carmel, Y. Maarek, A. Soffer, and R. Lempel. Knowledge encapsulation for focused search from pervasive devices. In *WWW'01: Proceedings of the 10th international conference on World Wide Web*, pages 754–764, New York, NY, USA, 2001. ACM.

[67] F. Arman, R. Depommier, A. Hsu, and M.-Y. Chiu. Content-based browsing of video sequences. In *Proceedings of ACM Multimedia*, pages 97–103, 1994.

[68] W. Y. Arms. Implementing Policies for Access Management. *D-Lib Magazine*, February 1998. http://www.dlib.org/dlib/february98/arms/02arms.html.

[69] W. Y. Arms. *Digital Libraries*. MIT Press, 2000.

[70] S. Arnold. How Google's Internet search is transforming application software. In *The Google Legacy*, pages 169–188. Infonortics, 2005.

[71] G. Arocena and A. Mendelzon. WebOQL: Restructuring documents, databases and Webs. In *Int. Conf. on Data Engineering*, pages 24–33, Orlando Florida, 1998.

[72] G. O. Arocena, A. O. Mendelzon, and G. A. Mihaila. Applications of a Web query language. In *Proc. 6th. Int'l. WWW Conf.*, Apr. 1997.

[73] P. Arvola, M. Junkkari, and J. Kekäläinen. Generalized contextualization method for XML information retrieval. In *ACM CIKM International Conference on Information and Knowledge Management, Bremen, Germany*, pages 20–27, 2005.

[74] E. Ashoori, M. Lalmas, and T. Tsikrika. Examining topic shifts in content-oriented XML retrieval. *International Journal on Digital Libraries*, 8(1):39–60, 2007.

[75] Ask MyStuff. urlhttp://about.ask.com/en/docs/mystuff/tour.shtml.

[76] Ask.com. Advanced search tips. http://help.ask.com/en/docs/about/adv_search_tips.shtml.

[77] J. A. Aslam and V. Pavlu. Query hardness estimation using Jensen-Shannon divergence among multiple scoring functions. In *Advances in Information Retrieval: 28th European Conference on IR Research*, pages 198–209, 2007.

[78] R. Attar and A. Fraenkel. Local feedback in full-text retrieval systems. *Journal of the ACM*, 24(3):397–417, 1977.

[79] G. Attardi and M. Ciaramita. Tree revision learning for dependency parsing. In C. L. Sidner, T. Schultz, M. Stone, and C. Zhai, editors, *HLT-NAACL*, pages 388–395, Rochester, NY, USA, April 2007. The Association for Computational Linguistics.

[80] A. Aula. Enhancing the readability of search result summaries. *Proceedings of HCI 2004*, pages 6–10, 2004.

[81] A. Aula. *Studying user strategies and characteristics for developing Web search interfaces.* PhD thesis, University of Tampere, Finland, Ph.D. Dissertation, Dissertations in Interactive Technology, Number 3., 2005.

[82] S. Axelrod, V. Goel, R. A. Gopinath, P. Olsen, and K. Visweswariah. Subspace constrained Gaussian mixture models for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 13(6):1144–1160, November 2005.

[83] C. Badue, R. Baeza-Yates, B. A. Ribeiro-Neto, A. Ziviani, and N. Ziviani. Analyzing imbalance among homogeneous index servers in a Web search system. *Information Processing & Management*, 43(3), 2007.

[84] C. S. Badue, J. Almeida, V. Almeida, R. Baeza-Yates, B. A. Ribeiro-Neto, A. Ziviani, and N. Ziviani. Capacity planning for vertical search engines. Submitted for publication, 2009.

[85] C. S. Badue, R. Baeza-Yates, B. A. Ribeiro-Neto, A. Ziviani, and N. Ziviani. Modeling performance-driven workload characterization of Web search systems. In P. S. Yu, V. J. Tsotras, E. A. Fox, and B. Liu, editors, *CIKM*, pages 842–843, Arlington, Virginia, USA, November 2006. ACM.

[86] R. Baeza-Yates. Challenges in the interaction of information retrieval and natural language processing. In A. F. Gelbukh, editor, *5th Int. Conf. on Computational Linguistics and Intelligent Text Processing*, volume 2945 of *Lecture Notes in Computer Science*, pages 445–456, Seoul, South Korea, November 2004. Springer.

[87] R. Baeza-Yates. A fast set intersection algorithm for sorted sequences. In S. C. Sahinalp, S. Muthukrishnan, and U. Dogrusöz, editors, *CPM*, volume 3109 of *Lecture Notes in Computer Science*, pages 400–408, Istanbul,Turkey, 2004. Springer.

[88] R. Baeza-Yates. Query usage mining in search engines. *Web Mining: Applications and Techniques, Anthony Scime, editor. Idea Group*, 2004.

[89] R. Baeza-Yates. Applications of Web query mining. *European Conference on Information Retrieval (ECIR'05), D. Losada, J. Fernández-Luna (editors), Springer LNCS 3408*, pages 7–22, 2005.

[90] R. Baeza-Yates. Graphs from search engine queries. In J. van Leeuwen, G. F. Italiano, W. van der Hoek, C. Meinel, H. Sack, and F. Plasil, editors, *SOFSEM: Theory and Practice of Computer Science*, volume 4362 of *Lecture Notes in Computer Science*, pages 1–8, Harrachov, Czech Republic, January 2007. Springer.

[91] R. Baeza-Yates, P. Boldi, and C. Castillo. Generalizing PageRank: Damping functions for link-based ranking algorithms. In *Proceedings of SIGIR*, Seattle, Washington, USA, August 2006. ACM Press.

[92] R. Baeza-Yates, P. Boldi, and C. Castillo. Generic damping functions for propagating importance in link-based ranking. *Internet Mathematics*, 3(4):445–478, 2006.

[93] R. Baeza-Yates, L. Calderón-Benavides, and C. González-Caro. The intention behind Web queries. In F. Crestani, P. Ferragina, and M. Sanderson, editors, *Proceedings of String Processing and Information Retrieval (SPIRE)*, volume 4209 of *Lecture Notes in Computer Science*, pages 98–109. Springer, 2006.

[94] R. Baeza-Yates and C. Castillo. Relating Web characteristics with link based web page ranking. In *Proceedings of String Processing and Information Retrieval SPIRE*, pages 21–32, Laguna San Rafael, Chile, 2001. IEEE CS Press.

[95] R. Baeza-Yates and C. Castillo. Balancing volume, quality and freshness in Web crawling. In *Soft Computing Systems - Design, Management and Applications*, pages 565–572, Santiago, Chile, 2002. IOS Press Amsterdam.

[96] R. Baeza-Yates and C. Castillo. Crawling the infinite web: five levels are enough. *Journal of Web Engineering*, 6:49–72, 2007.

[97] R. Baeza-Yates, C. Castillo, and E. Efthimiadis. Characterization of national Web domains. *ACM TOIT*, 7(2), 2007.

[98] R. Baeza-Yates, C. Castillo, and F. S. Jean. Web dynamics, structure and page quality. In M. Levene and A. Poulovassilis, editors, *Web Dynamics*, pages 93–109. Springer, 2004.

[99] R. Baeza-Yates, C. Castillo, F. Junqueira, V. Plachouras, and F. Silvestri. Challenges on distributed Web retrieval. In *Proceedings of ICDE 2007*, pages 6–20. IEEE, 2007.

[100] R. Baeza-Yates, C. Castillo, M. Marín, and A. Rodríguez. Crawling a country: Better strategies than breadth-first for Web page ordering. In *Proceedings of the 14th international conference on World Wide Web*, pages 864–872, Chiba, Japan, 2005. ACM Press.

[101] R. Baeza-Yates, M. Ciaramita, P. Mika, and H. Zaragoza. Towards semantic search. In E. Kapetanios, V. Sugumaran, and M. Spiliopoulou, editors, *Natural Language and Information Systems, 13th International Conference on Applications of Natural Language to Information Systems, NLDB 2008*, volume 5039 of *Lecture Notes in Computer Science*, pages 4–11, London, UK, June 2008. Springer.

[102] R. Baeza-Yates, W. Cunto, U. Manber, and S. Wu. Proximity matching using fixed-queries trees. In *Proc. of Combinatorial Pattern Matching*, number 807 in LNCS, pages 198–212. Springer-Verlag, 1994.

[103] R. Baeza-Yates, N. Fuhr, and Y. Maarek. Second edition of the XML and information retrieval workshop held at sigir'2002. *SIGIR Forum*, 36(2):53–57, 2002.

[104] R. Baeza-Yates, A. Gionis, F. Junqueira, V. Murdock, V. Plachouras, and

F. Silvestri. The Impact of Caching on Search Engines. In *SIGIR'07: Proceedings of the 30th International ACM SIGIR conference on Research and Development in Information Retrieval*, Amsterdam, The Netherlands, 2007.

[105] R. Baeza-Yates, A. Gionis, F. Junqueira, V. Murdock, V. Plachouras, and F. Silvestri. Design trade-offs for search engine caching. *TWEB*, 2(4), 2008.

[106] R. Baeza-Yates, A. Gionis, F. Junqueira, V. Plachouras, and L. Telloli. On the feasibility of multi-site Web search engines. In *ACM CIKM 2009*, pages 425–434, Hong Kong, China, November 2009.

[107] R. Baeza-Yates and G. Gonnet. Efficient Text Searching of Regular Expressions. In G. Ausiello, M. Dezani-Ciancaglini, and S. R. D. Rocca, editors, *ICALP'89*, number 372 in LNCS, pages 46–62, Stresa, Italy, 1989. Springer-Verlag.

[108] R. Baeza-Yates and G. Gonnet. Fast text searching for regular expressions or automaton searching on a trie. *J. of the ACM*, 43(6):915–936, Nov 1996.

[109] R. Baeza-Yates, C. Hurtado, and M. Mendoza. Query clustering for boosting Web page ranking. *Advances in Web Intelligence, AWIC 2004, Springer LNCS*, 3034:164–175, 2004.

[110] R. Baeza-Yates, C. Hurtado, and M. Mendoza. Query recommendation using query logs in search engines. In *Current Trends in Database Technology - EDBT*, volume 3268, pages 588–596. Springer-Verlag GmbH, 2004.

[111] R. Baeza-Yates, C. A. Hurtado, and M. Mendoza. Improving search engines by query clustering. *JASIST*, 58(12):1793–1804, 2007.

[112] R. Baeza-Yates, F. Junqueira, V. Plachouras, and H. F. Witschel. Admission Policies for Caches of Search Engine Results. In *SPIRE'07: Proceedings of the 14th Symposium on String Processing and Information Retrieval*, Santiago, Chile, 2007.

[113] R. Baeza-Yates, A. Moffat, and G. Navarro. Searching large text collections. In J. Abello, P. M. Pardalos, and M. G. C. Resende, editors, *Handbook of Massive Data Sets*, pages 195–244. Kluwer Academic Publishers, 2002.

[114] R. Baeza-Yates, V. Murdock, and C. Hauff. Efficiency trade-offs in two-tier Web search systems. In J. Allan, J. A. Aslam, M. Sanderson, C. Zhai, and J. Zobel, editors, *SIGIR*, pages 163–170, Boston, MA, USA, 2009. ACM.

[115] R. Baeza-Yates and G. Navarro. Integrating contents and structure in text retrieval. *ACM SIGMOD Record*, 25(1):67–79, Mar. 1996.

[116] R. Baeza-Yates and G. Navarro. Block-addressing indices for approximate text retrieval. In *Proc. of the 6th CIKM Conference*, pages 1–8, Las Vegas, Nevada, 1997.

[117] R. Baeza-Yates and G. Navarro. Faster approximate string matching. *Algorithmica*, 23(2):127–158, 1999.

[118] R. Baeza-Yates, G. Navarro, J. Vegas, and P. de la Fuente. A model and a visual query language for structured text. In B. A. Ribeiro-Neto, editor, *Proc. of the 5th Symposium on String Processing and Information Retrieval*, pages 7–13, Santa Cruz, Bolivia, Sept 1998. IEEE CS Press.

[119] R. Baeza-Yates, N.Fuhr, and Y. Maarek. Introduction to the special issue on XML retrieval. *ACM Transactions on Information Systems*, 24(4):405–406, 2006.

[120] R. Baeza-Yates, A. Pereira, and N. Ziviani. Genealogical trees on the Web: A search engine user perspective. In *WWW'08: Proceedings of the 17th international conference on World Wide Web*, pages 367–376, Beijing, China, 2008.

[121] R. Baeza-Yates and B. Poblete. Dynamics of the Chilean Web structure. In *Proceedings of the 3rd International Workshop on Web Dynamics*, New York, USA, 2004.

[122] R. Baeza-Yates and B. Poblete. A website mining model centered on user queries. In *Semantics, Web and Mining, Joint International Workshops, EWMF 2005 and KDO 2005*, volume 4289 of *Lecture Notes in Computer Science*, pages 1–17, Porto, Portugal, October 2005. Springer.

[123] R. Baeza-Yates and F. Saint-Jean. A three level search engine index based in query log distribution. In M. A. Nascimento, E. S. de Moura, and A. L. Oliveira, editors, *SPIRE*, volume 2857 of *Lecture Notes in Computer Science*, pages 56–65, Manaus, Brazil, October 2003. Springer.

[124] R. Baeza-Yates and A. Salinger. Experimental analysis of a fast intersection algorithm for sorted sequences. In M. P. Consens and G. Navarro, editors, *SPIRE*, volume 3772 of *Lecture Notes in Computer Science*, pages 13–24, Buenos Aires, Argentina, November 2005. Springer.

[125] R. Baeza-Yates and A. Tiberi. Extracting semantic relations from query logs. In P. Berkhin, R. Caruana, and X. Wu, editors, *KDD*, pages 76–85, San Jose, CA, USA, 2007. ACM.

[126] P. Bailey, N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec 2007 enterprise track. In *The Sixteenth Text REtrieval Conference (TREC 2007) Proceedings*, Gaithersburg, MD, 2007. NIST. TREC Special Publication: SP 500-274. trec.nist.gov/pubs/trec16/papers/ENT.OVERVIEW16.pdf.

[127] P. Bailey, D. Hawking, and B. Matson. Secure search in enterprise webs: Tradeoffs in efficient implementation for document level security. In *Proceedings of CIKM 2006*, 2006. http://es.csiro.au/pubs/cikm127_bailey.pdf.

[128] D. Bainbridge, J. Thompson, and I. H. Witten. Assembling and enriching digital library collections. In *JCDL'03: Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 323–334, Houston, Texas, 2003.

[129] J. Baker. UCLA-NSF Social Aspects of Digital Libraries Workshop, January 1996. http://www.gslis.ucla.edu/DL/.

[130] P. Baldi, P. Frasconi, and P. Smyth. *Modeling the Internet and the Web: Probabilistic Methods and Algorithms*. John Wiley & Sons, May 2003.

[131] K. Balog, L. Azzopardi, and M. de Rijke. A language modeling framework for expert finding. *Information Processing and Management*, 45(1):1–19, January 2009. doi:10.1016/j.ipm.2008.06.003.

[132] K. Balog and M. de Rijke. Combining candidate and document models for expert search. In *The Seventeenth Text Retrieval Conference (TREC 2008)*. NIST, 2009. Special Publication.

[133] S. Baluja and M. Covell. Learning 'forgiving' hash functions: Algorithms and large-scale tests. In *International Joint Conference on AI*, January 2007.

[134] S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly. Video suggestion and discovery for YouTube: Taking random walks through the view graph. In *WWW'08: Proceeding of the 17th International Conference on World Wide Web*, pages 895–904, New York, NY, USA, 2008. ACM.

[135] Z. Bar-Yossef and M. Gurevich. Random sampling from a search engine's index. In *WWW'06: Proceedings of the 15th international conference on World Wide Web*, pages 367–376, New York, NY, USA, 2006. ACM Press.

[136] Z. Bar-Yossef and M. Gurevich. Efficient search engine measurements. In C. L. Williamson, M. E. Zurko, P. F. Patel-Schneider, and P. J. Shenoy, editors, *WWW*, pages 401–410, Banff, Canada, 2007. ACM.

[137] Z. Bar-Yossef and M. Gurevich. Mining search engine query logs via suggestion sampling. In *VLDB 2008*, 2008.

[138] Z. Bar-Yossef and M. Gurevich. Estimating the impression rank of Web pages. In *WWW 2009*, 2009.

[139] A.-L. Barabási. *Linked: the New Science of Networks*. Perseus Books Group, May 2002.

[140] A. L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, October 1999.

[141] M. Barbaro and T. Zeller. A face is exposed for AOL searcher no. 4417749. New York Times, 2006.

[142] J. Barbay and C. Kenyon. Adaptive intersection and t-threshold problems. In *SODA*, pages 390–399, 2002.

[143] J. Barbay, A. López-Ortiz, T. Lu, and A. Salinger. An experimental investigation of set intersection algorithms for text searching. *Journal of Experimental Algorithms (JEA)*, 14(3):7–24, 2009.

[144] L. Barbosa, F. Junqueira, V. Plachouras, and R. Baeza-Yates. Variability as a measure a search quality, 2009. Submitted.

[145] R. Barbosa. *Query Performance on Distributed Digital Libraries*. CS Department, Federal University of Minas Gerais, Brazil, 1998. Master Thesis. In Portuguese.

[146] P. Barford and M. Crovella. Generating representative Web workloads for network and server performance evaluation. In *ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, pages 151–160, July 1998.

[147] H. Barlow. Unsupervised learning. *Neural Computation*, 1(3):295–311, 1989.

[148] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, and M. Jordan. Matching words and pictures. *Special Issue on Text and Images, Journal of Machine Learning Research*, 2002.

[149] J. Baron, D. Lewis, and D. Oard. TREC-2006 Legal Track Overview. In *Proceedings of TREC 2006*, pages 79–98. NIST, 2007. http://trec.nist.gov/pubs/trec15/t15_proceedings.html.

[150] D. Barreau and B. Nardi. Finding and Reminding: File Organization From the Desktop. *ACM SIGCHI Bulletin*, 27(3):39–43, 1995.

[151] L. A. Barroso, J. Dean, and U. Hölzle. Web search for a planet: the Google Cluster Architecture. *IEEE Micro Magazine*, 23(2):22–28, Mar./Apr. 2003.

[152] L. A. Barroso and U. Hölzle. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*, volume 6 of *Synthesis Lectures on Computer Architecture*. Morgan Claypool, 2009.

[153] B. T. Bartell, G. W. Cottrell, and R. K. Belew. Latent semantic indexing is an optimal special case of multidimensional scaling. In *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Information Retrieval Theory, pages 161–167, 1992.

[154] M. Bartsch and G. Wakefield. To catch a chorus: Using chroma-based representations for audio thumbnailing. In *2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, pages 15–18, Octobter 2001.

[155] N. A. Basbanes. Foreword. The Library – An Illustrated History, authored by Stuart A.P. Murray, published by Skyhorse Publishing, 2009.

[156] M. Bates. Information search tactics. *Journal of the American Society for Information Science*, 30(4):205–214, 1979.

[157] M. Bates. The design of browsing and berrypicking techniques for the on-line search interface. *Online Review*, 13(5):407–431, 1989.

[158] M. Bates. Where should the person stop and the information search interfaces start? *Information Processing and Management*, 26(5), 1990.

[159] M. Bates. Improving user access to library catalog and portal information. Task force recommendation 2.3, final report (version 3), Library of Congress Bicentennial Conference on Bibliographic Control for the New Millenium, 2003. http://www.loc.gov/catdir/bibcontrol/2.3BatesReport6-03.doc.pdf.

[160] P. Baudisch, B. Lee, and L. Hanna. Fishnet, a fisheye Web browser with search term popouts: a comparative evaluation with overview and linear view. *Proceedings of the working conference on Advanced Visual Interfaces (AVI'04)*, pages 133–140, 2004.

[161] H. Bay, T. Tuytelaars, and L. J. V. Gool. Surf: Speeded up robust features. In *ECCV (1)*, pages 404–417, 2006.

[162] D. Bearman. Digital Libraries. In B. Cronin, editor, *Annual Review of Information Science and Technology*, volume 41, pages 223–272. American Society for Information Science and Technology, 2007.

[163] M. Beaudouin-Lafon and W. Mackay. Prototyping Tools and Techniques. In *Human-Computer Interaction Handbook*. Lawrence Erlrbaum Associates, 2003.

[164] J. Becker and R. Hayes. *Information storage and retrieval: tools, elements, theories*. Wiley, 1963.

[165] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and rectangles. *ACM SIGMOD*, pages 322–331, May 1990.

[166] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 407–416. ACM, 2000.

[167] C. Beeri and Y. Kornatzky. A logical query language for hypertext systems. In *Proc. of the European Conference on Hypertext*, pages 67–80. Cambridge University Press, 1990.

[168] S. M. Beitzel, E. C. Jensen, A. Chowdhury, D. Grossman, and O. Frieder. Hourly analysis of a very large topically categorized Web query log. In *Proceedings of the 27th annual international conference on Research and development in information retrieval*, pages 321–328, Sheffield, United Kingdom, 2004. ACM Press.

[169] S. M. Beitzel, E. C. Jensen, O. Frieder, D. A. Grossman, D. D. Lewis, A. Chowdhury, and A. Kolcz. Automatic Web query classification using labeled and unlabeled training data. In R. Baeza-Yates, N. Ziviani, G. Marchionini, A. Moffat, and J. Tait, editors, *SIGIR*, pages 581–582, Salvador, Brazil, August 2005. ACM.

[170] R. Belew. *Finding Out About: A Cognitive Perspective on Search Engine Technology and the WWW*. Cambridge University Press, 2000.

[171] N. Belkin, D. Kelly, G. Kim, J. Kim, H. Lee, G. Muresan, M. Tang, X. Yuan, and C. Cool. Query length in interactive information retrieval. *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'03)*, pages 205–212, 2003.

[172] R. M. Bell, Y. Koren, and C. Volinsky. Chasing $1,000,000: How we won the Netflix progress prize. *Statistical Computing and Statistical Graphics Newsletter*, 18:4–12, 2007.

[173] T. Bell, J. Cleary, and I. Witten. Data compression using adaptive coding and partial string matching. *IEEE Trans. on Communications*, 32(4):396–402, 1984.

[174] T. Bell, J. Cleary, and I. Witten. *Text Compression*. Prentice-Hall, 1990.

[175] T. C. Bell, A. Moffat, C. Nevill-Manning, I. H. Witten, and J. Zobel. Data compression in full-text retrieval systems. *Journal of the American Society for Information Science*, 44:508–531, 1993.

[176] A. Belussi and C. Faloutsos. Estimating the selectivity of spatial queries using the 'correlation' fractal dimension. In *Proc. of VLDB Conf.*, pages 299–310, Zurich, Switzerland, Sept. 1995.

[177] Y. Ben-Aharon, S. Cohen, Y. Grumbach, Y. Kanza, J. Mamou, Y. Sagiv, B. Sznajder, and E. Twito. Searching in an XML corpus using content and structure. In *INEX 2003 Proceedings*, pages 46–52, 2003.

[178] I. Ben-Shaul, M. Herscovici, M. Jacovi, Y. Maarek, D. Pelleg, M. Shtalhaim, V. Soroka, and S. Ur. Adding support for dynamic and focused search with fetuccino. *Computer Networks*, 31(11-16):1653–1665, 1999. Also appeared in the Proceedigns of WWW8.

[179] M. Bender, S. Michel, P. Triantafillou, G. Weikum, and C. Zimmer. Improving Collection Selection With Overlap Awareness in P2P Search Engines. In *SIGIR'05: Proceedings of the 28th International ACM SIGIR conference on Research and Development in Information Retrieval*, Salvador, Brazil, 2005.

[180] M. Bender, S. Michel, P. Triantafillou, G. Weikum, and C. Zimmer. MINERVA: collaborative P2P search. In *VLDB'05: Proceedings of the 31st International conference on Very Large Data Bases*, Trondheim, Norway, 2005.

[181] M. Bender, S. Michel, P. Triantafillou, G. Weikum, and C. Zimmer. P2P content search: Give the Web back to the people. International Workshop on Peer-to-Peer Systems (IPTPS), February 2006.

[182] Y. Benkler. Coase's penguin, or, Linux and the nature of the firm. *Yale Law Journal*, 112:371–446, 2002.

[183] P. N. Bennett, S. T. Dumais, and E. Horvitz. The combination of text classifiers using reliability indicators. *Inf. Retr.*, 8(1):67–100, 2005.

[184] J. Bentley, D. Sleator, R. Tarjan, and V. Wei. A locally adaptive data compression scheme. *Communications of the ACM*, 29(4):320–330, apr 1986.

[185] S. Berchtold, D. A. Keim, and H.-P. Kriegel. The X-tree : An index structure for high-dimensional data. *VLDB*, pages 28–39, 1996.

[186] T. L. Berg, A. C. Berg, J. Edwards, and D. A. Forsyth. Who's in the picture In *Proceedings of the Neural Information Processing Society*, 2004.

[187] A. Berger and J. Lafferty. Information retrieval as a statistical translation In *ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 222–229, 1999.

[188] D. Bergmark. Collection synthesis. In *Proceedings of the Second ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 253–262, Portland, OR, 2002.

[189] P. Berkhin. A survey on pagerank computing. *Internet Mathematics*, 2(2):73–120, 2005.

[190] T. Berners-Lee. The World Wide Web Consortium. http://www.w3.org.

[191] T. Berners-Lee. Universal resource identifiers in WWW. RFC 1630. http://www.w3.org/Addressing/rfc1630.txt, 1994.

[192] T. Berners-Lee, R. Cailliau, A. Luotonen, H. F. Nielsen, and A. Secret. The World-Wide Web. *Comm. of the ACM*, 37(8):76–82, 1994.

[193] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform resource identifiers (uri): Generic syntax. RFC 2396. http://www.ietf.org/rfc/rfc2396.txt, 1998.

[194] M. Berry and M. Browne. *Understanding Search Engines – Mathematical Mod-*

eling and Text Retrieval. Siam, 2005.

[195] S. Betsi, M. Lalmas, A. Tombros, and T. Tsikrika. User expectations from XML element retrieval. In *29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA*, pages 611–612, 2006.

[196] K. Bharat. Searchpad: explicit capture of search context to support Web search. In *Proceedings of the 9th international World Wide Web conference on Computer networks : the international journal of computer and telecommunications networking*, pages 493–501, Amsterdam, the Netherlands, the Netherlands, 2000. North-Holland Publishing Co.

[197] K. Bharat, A. Broder, M. Henzinger, P. Kumar, and S. Venkatasubramanian. The connectivity server: fast access to linkage information on the Web. In *7th WWW Conf.*, Brisbane, Australia, April 1998.

[198] K. Bharat and A. Z. Broder. A technique for measuring the relative size and overlap of public Web search engines. In *7th WWW Conference*, pages 379–388, Brisbane, Australia, 1998.

[199] K. Bharat and M. R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 104–111, Melbourne, Australia, August 1998. ACM Press, New York.

[200] M. Bianchini, M. Gori, and F. Scarselli. Inside pagerank. *ACM Trans. Inter. Tech.*, 5(1):92–128, February 2005.

[201] D. Bilal. Children's use of the Yahooligans! Web search engine. I. Cognitive, physical, and affective behaviors on fact-based search tasks. *Journal of the American Society for Information Science*, 51(7):646–665, 2000.

[202] Bing. Advanced search keywords. http://help.live.com/Help. aspx?market=en-US&project=WL_Searchv1&querytype=topic&query= WL_SEARCH_REF_Keywords.htm.

[203] Bing. Use advanced search. http://help.live.com/help.aspx?mkt=en-us&project=wl_searchv1&querytype=keyword&query=redliub&tmt=&domain=www.bing.com:80.

[204] J. Bing. *Handbook of Legal Information Retrieval.* Elsevier Science Inc., New York, NY, USA, 1984.

[205] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2007.

[206] R. Blanco and A. Barreiro. Document identifier reassignment through dimensionality reduction. In *Advances in Information Retrieval: 27th European Conference on IR research, ECIR 2005, Santiago de Compostela, Spain, March 21-23, 2005. Proceedings*, pages 375 – 387, 2005.

[207] A. Blandford, S. Keith, I. Connell, and H. Edwards. Analytical usability evaluation for digital libraries: a case study. In *Proc. of JCDL'04*, pages 27 36, Tucson, AZ, 2004.

[208] A. Blandford, H. Stelmaszewska, and N. Bryan-Kinns. Use of multiple digital libraries: A case study. In *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 179–188, Roanoke, Virginia, 2001.

[209] D. Blandford and G. Blelloch. Index compression through document reordering. In *Proceedings of the Data Compression Conference (DCC'02)*, pages 342–351, Washington, DC, USA, 2002. IEEE Computer Society.

[210] H. Blanken, T. Grabs, H.-J. Schek, R. Schenkel, and G. Weikum, editors. *Intelligent Search on XML Data, Applications, Languages, Models, Implementations, and Benchmarks*, volume 2818. Springer, 2003.

[211] D. M. Blei and M. I. Jordan. Modeling annotated data. In *SIGIR '03: Pro-*

ceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval, pages 127–134, New York, NY, USA, 2003. ACM.

[212] J. F. Blinn. What's that deal with the DCT? *Computer Graphics and Applications*, 13:78–83, July 1993.

[213] B. H. Bloom. Space/Time Trade-offs in Hash Coding with Allowable Errors. *Communications of the ACM*, 13(7), 1970.

[214] A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, 1997.

[215] J. Blustein, I. Ahmed, and K. Instone. An evaluation of look-ahead breadcrumbs for the WWW. In S. Reich and M. Tzagarakis, editors, *Hypertext*, pages 202–204, Salzburg, Austria, September 2005. ACM.

[216] Budapest open access initiative, 2001. http://www.soros.org/openaccess/.

[217] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna. The query-flow graph: model and applications. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 609–618, Napa Valley, California, USA, 2008. ACM.

[218] P. Boldi, F. Bonchi, C. Castillo, and S. Vigna. From "Dango" to "Japanese Cakes": Query reformulation models and patterns. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 183–190, Milano, Italy, 2009. IEEE CS Press.

[219] P. Boldi, B. Codenotti, M. Santini, and S. Vigna. Ubicrawler: a scalable fully distributed Web crawler. *Software, Practice and Experience*, 34(8):711–726, 2004.

[220] P. Boldi, M. Santini, and S. Vigna. Do your worst to make the best: Paradoxical effects in pagerank incremental computations. In *Proceedings of the third Workshop on Web Graphs (WAW)*, volume 3243 of *Lecture Notes in Computer Science*, pages 168–180, Rome, Italy, October 2004. Springer.

[221] P. Boldi, M. Santini, and S. Vigna. Pagerank as a function of the damping factor. In *Proceedings of the 14th international conference on World Wide Web*, pages 557–566, Chiba, Japan, 2005. ACM Press.

[222] P. Boldi and S. Vigna. The webgraph framework I: Compression techniques. In S. I. Feldman, M. Uretsky, M. Najork, and C. E. Wills, editors, *Proceedings of the 13th conference on World Wide Web*, pages 595–602, New York, NY, USA, 2004. ACM Press.

[223] J. Bollen, R. Luce, S. S. Vemulapalli, and W. Xu. Usage analysis for the identification of research trends in digital libraries. *D-Lib Magazine*, 9, 2003.

[224] M. Bolsky and D. Korn. *The New KornShell Command and Programming Language*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1995.

[225] C. Bonacic, C. García, M. Marín, M. Prieto, and F. Tirado. Exploiting Hybrid Parallelism in Web Search Engines. In *14th European International Conference on Parallel Processing (Euro-Par 2008)*, LNCS 5168, pages 414–423, Las Palmas de Gran Canaria, Spain, August 2008.

[226] P. Bonnet and A. Tomasic. Partial answers for unavailable data sources. In *Workshop on Flexible Query-Answering Systems*, pages 43–54, 1998.

[227] A. Bookstein. On the perils of merging Boolean and weighted retrieval systems. *Journal of the American Society for Information Sciences*, 29(3):156–158, 1978.

[228] A. Bookstein. Fuzzy requests: An approach to weighted Boolean searches. *Journal of the American Society for Information Sciences*, 31:240–247, 1980.

[229] A. Bookstein. Implication of Boolean structure for probabilistic retrieval. In *Proc of the Eight Annual International ACM/SIGIR Conference on Research*

and Development in Information Retrieval, pages 11–17, Montreal, Canada, 1985.

[230] J. Boreczky, A. Girgensohn, G. Golovchinsky, and S. Uchihashi. An interactive comic book presentation for exploring video. In *CHI '00: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 185–192, New York, NY, USA, 2000. ACM.

[231] J. Boreczky and L. Rowe. Comparison of video shot boundary detection techniques. In *IS&T SPIE Symposium on Electronic Imaging*, volume 2670, pages 170–179, San Jose, 1996.

[232] C. Borgman. Why are online catalogs *still* hard to use? *Journal of the American Society for Information Science*, 47(7):493–503, 1996.

[233] C. L. Borgman. Social aspects of digital libraries. In *DL'96: Proceedings of the 1st ACM International Conference on Digital Libraries*, D-Lib Working Session 2A, pages 170–171, 1996.

[234] C. L. Borgman. What are digital libraries? competing visions. *Inf. Process. Manage.*, 35(3):227-243, 1999.

[235] C. L. Borgman, G. Leazer, A. J. Gilliland-Swetland, K. Millwood, C. Champeny, J. Finley, and L. J. Smart. How geography professors select materials for classroom lectures: Implications for the design of digital libraries. In *Proc. of JCDL'04*, pages 179–185, Tucson, AZ, 2004.

[236] C. L. Borgman, G. H. Leazer, A. J. Gilliland-Swetland, and R. Gazan. Iterative design and evaluation of a geographic digital library for university students: A case study of the Alexandria Digital Earth Prototype (ADEPT). *LNCS*, 2163:390, 2001.

[237] P. Borlund. The IIR evaluation model: a framework for evaluation of interactive information retrieval systems. *Information Research*, 8(3), 2003.

[238] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Finding authorities and hubs from link structures on the world wide web. In *WWW'01: Proceedings of the 10th international conference on World Wide Web*, pages 415–429, New York, NY, USA, 2001. ACM Press.

[239] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Link analysis ranking: algorithms, theory, and experiments. *ACM Trans. Inter. Tech.*, 5(1):231-297, February 2005.

[240] D. Borthakur. *The Hadoop Distributed File System: Architecture and Design.* The Apache Software Foundation, 2007.

[241] J. Bosak. XML, Java, and the future of the Web. Technical report, Sun Microsystems, 1997. http://sunsite.unc.edu/pub/sun-info/standards/xml/why/xmlapps.htm.

[242] A. Bosch, A. Zisserman, and X. Munoz. Scene classification via pLSA. In *European Conference on Computer Vision*, 2006.

[243] C. P. Bourne and T. B. Hahn. *A history of online information services, 1963-1976.* MIT Press, Cambridge, Mass., 2003.

[244] C. M. Bowman, P. B. Danzig, D. R. Hardy, U. Manber, and M. F. Schwartz. The Harvest information discovery and access system. In *Proc. 2nd Inter. World Wide Web Conf.*, pages 763–771, Oct. 1994.

[245] J. Boyan, D. Freitag, and T. Joachims. A machine learning architecture for optimizing web search engines. In *AAAI Workshop on Internet Based Information Systems*, pages 1 – 8, August 1996.

[246] R. S. Boyer and J. S. Moore. A fast string searching algorithm. *Communications of the ACM*, 20(10):762–772, 1977.

[247] T. Bozkaya and Z. M. Ozsoyoglu. Distance-based indexing for high-dimensional metric spaces. In *Proc. of ACM SIGMOD Conference*, pages 357–368, Tucson, AZ, USA, 1997.

[248] P. D. Bra and R. Post. Searching for arbitrary information in the WWW: the fish search for Mosaic. In *Proc. of the Second International World Wide Web Conference*, Chicago, Oct. 1994. http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/www-fall94.html.

[249] A. Bratko and B. Filipic. Exploiting structural information for semi-structured document categorization. *Information Processing and Management*, 42(3):679–694, 2006.

[250] T. Bray. Measuring the web. In *Fifth International World Wide Web Conference*, Paris, May 1996. http://www5conf.inria.fr/fich_html/papers/P9/Overview.html.

[251] M. Breaks. The eLib Hybrid Library Projects. *Ariadne*, (28), 2001. http://www.ariadne.ac.uk/issue28/hybrid/.

[252] M. Breeding. Musings on the state of the ILS in 2006. *Computers in Libraries*, 26(26):26–29, 2006.

[253] M. Breeding. Making a business case for open source ILS. *Computers in Libraries*, 28(28):36–39, 2008.

[254] M. Breeding. Open source integrated library systems. *Library Technology Reports*, 44(8), 2008.

[255] M. Breeding. Opportunity out of turmoil. *Library Journal*, 133(6):32, 2008.

[256] L. Breiman. Stacked regressions. *Machine Learning*, 24:49–64, 1996.

[257] B. Brewington and G. Cybenko. Keeping up with the changing web. *IEEE Computer*, 33(5):52–58, May 2000.

[258] B. Brewington, G. Cybenko, R. Stata, K. Bharat, and F. Maghoul. How dynamic is the web? In *Proceedings of the Ninth Conference on World Wide Web*, Amsterdam, Netherlands, May 2000. ACM Press.

[259] Bright-Planet. Deep Web white paper. Available online at brightplanet.com, July 2000.

[260] S. Brin. Near neighbor search in large metric spaces. In *Proc. of VLDB Conf.*, pages 574–584, Zurich, Switzerland, Sept 1995.

[261] S. Brin. Extracting patterns and relations from the World Wide Web. In *Workshop on Web Databases*, Valencia, Spain, March 1998.

[262] S. Brin, J. Davis, and H. Garcia-Molina. Copy detection mechanisms for digital documents. In M. J. Carey and D. A. Schneider, editors, *SIGMOD Conference*, pages 398–409, San Jose, CA, USA, 1995. ACM Press.

[263] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. In *WWW7: Proceedings of the Seventh International Conference on World Wide Web 7*, pages 107–117. Elsevier Science Publishers B. V., 1998.

[264] T. Brinkhoff, H.-P. Kriegel, R. Schneider, and B. Seeger. Multi-step processing of spatial joins. In *Proc. of ACM SIGMOD*, pages 197–208, Minneapolis, MN, USA, May 1994.

[265] N. Brisaboa, A. Fariña, G. Navarro, and M. Esteller. (s,c)-dense coding: An optimized compression code for natural language text databases. In *Proceedings of the 10th International Symposium on String Processing and Information Retrieval (SPIRE 2003)*, LNCS 2857, pages 122–136. Springer, 2003.

[266] N. Brisaboa, A. Fariña, G. Navarro, and J. Paramá. Efficiently decodable and

searchable natural language adaptive compression. In *Proc. 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'05)*, pages 234–241. ACM Press, 2005.

[267] A. Broder. On the resemblance and containment of documents. In *SEQUENCES: Conf. on Compression and Complexity of Sequences*, pages 21–29, Salerno, Italy, 1997. IEEE Computer Society.

[268] A. Broder. A taxonomy of Web search. *ACM SIGIR Forum*, 36(2):3–10, 2002. http://www.acm.org/sigir/forum/F2002/broder.pdf.

[269] A. Broder, S. Glassman, M. Manasse, and G. Zweig. Syntactic clustering of the Web. In *6th Int'l WWW Conference*, pages 391–404, Santa Clara, CA, USA, April 1997.

[270] A. Broder and V. Josifovski. Introduction to computational advertising. Course at Stanford University, http://www.stanford.edu/class/msandc239/, Sept-Dec 2009.

[271] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web: Experiments and models. In *Proceedings of the Ninth Conference on World Wide Web*, pages 309–320, Amsterdam, Netherlands, May 2000. ACM Press.

[272] A. Z. Broder. The future of Web search: From information retrieval to information supply. In *NGITS*, page 362, 2006.

[273] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. Efficient query evaluation using a two-level retrieval process. In *Proceedings of CIKM 2003*, pages 426–434, New York, NY, USA, 2003. ACM Press.

[274] A. Z. Broder and A. C. Ciccolo. Towards the next generation of enterprise search technology. *IBM Syst. J.*, 43(3):451–454, 2004.

[275] A. Z. Broder, P. Ciccolo, M. Fontoura, E. Gabrilovich, V. Josifovski, and L. Riedel. Search advertising using Web relevance feedback. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 1013–1022, New York, NY, USA, 2008. ACM.

[276] A. Z. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. Robust classification of rare queries using Web knowledge. In W. Kraaij, A. P. de Vries, C. L. A. Clarke, N. Fuhr, and N. Kando, editors, *SIGIR*, pages 231–238, Amsterdam, The Netherlands, 2007 2007. ACM.

[277] A. Z. Broder, M. Fontoura, V. Josifovski, and L. Riedel. A semantic approach to contextual advertising. In W. Kraaij, A. P. de Vries, C. L. A. Clarke, N. Fuhr, and N. Kando, editors, *SIGIR*, pages 559–566, Amsterdam, The Netherlands, November 2007. ACM.

[278] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. In *Selected papers from the sixth international conference on World Wide Web*, pages 1157–1166, Essex, UK, 1997. Elsevier Science Publishers Ltd.

[279] A. Z. Broder and Y. S. Maarek, editors. *Proceedings of the SIGIR 2006 Workshop on Faceted Search*, Seattle, WA, USA, August 2006.

[280] J. Broglio, J. Callan, W. Croft, and D. Nachbar. Document retrieval and routing using the INQUERY system. In D. Harman, editor, *Overview of the Third Retrieval Conference (TREC-3)*, pages 29–38. NIST Special Publication 500-225, 1995.

[281] A. Broschart, R. Schenkel, M. Theobald, and G. Weikum. TopX @ INEX 2007. In *Focused access to XML documents, 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007*, Dagstuhl Castle, Germany, 2008. Selected Papers.

[282] E. W. Brown. *Execution Performance Issues in Full-Text Information Retrieval.* PhD thesis, University of Massachusetts, Amherst, 1996. Available as UMass Comp. Sci. Tech. Rep. TR95-81.

[283] S. Browne, J. Dongarra, J. Horner, P. McMahan, and S. Wells. Technologies for repository interoperation and access control. In *Proceedings of the 3rd ACM International Conference on Digital Libraries*, pages 40–48, 1998.

[284] R. Brunelli, O. Mich, and C. Modena. A survey on video indexing. Technical Report 9612-06, IRST, 1996.

[285] P. Bruza, R. McArthur, and S. Dennis. Interactive Internet Search: Keyword, Directory and Query Reformulation Mechanisms Compared. *Proceedings of the 23th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'00)*, pages 280–287, 2000.

[286] P. D. Bruza and T. P. Van Der Weide. Stratified hypermedia structures for information disclosure. *The Computer Journal*, 35(3):208–220, 1992.

[287] G. Buchanan, D. Bainbridge, K. J. Don, and I. H. Witten. A new framework for building digital library collections. In *Proceedings of the 5th ACM/IEEE Joint Conference on Digital Libraries (JCDL05)*, pages 23–31, Denver, CA, USA, 2005.

[288] G. Buchanan, S. Cunningham, A. Blandford, J. Rimmer, and C. Warwick. Information Seeking by Humanities Scholars. *Proceedings of the European Conference on Digital Libraries (ECDL'05)*, 2005.

[289] C. Buckley and G. Salton. Optimization of relevance feedback weights. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Seattle, Washington, USA, July 9-13, 1995 (Special Issue of the SIGIR Forum)*, pages 351–357, 1995.

[290] C. Buckley, G. Salton, and J. Allan. The effect of adding relevance information in a relevance feedback environment. In *Proc. of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 292–300, Dublin, Ireland, 1994.

[291] C. Buckley and E. M. Voorhees. Evaluating evaluation measure stability. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 33–40, 2000.

[292] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 25–32, 2004.

[293] N. Budhiraja, K. Marzullo, F. Schneider, and S. Toueg. Optimal primary-backup protocols. In *Proceedings of the International Workshop on Distributed Algorithms (WDAG)*, pages 362–378, Haifa, Israel, November 1992. Springer-Verlag.

[294] P. Buneman. Semistructured data. In *ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 117–121, Tucson, Arizona, 1997.

[295] C. J. C. Burges, R. Ragno, and Q. V. Le. Learning to rank with nonsmooth cost functions. In B. Schölkopf, J. C. Platt, T. Hoffman, B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *NIPS*, pages 193–200. MIT Press, 2006.

[296] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. N. Hullender. Learning to rank using gradient descent. In L. D. Raedt and S. Wrobel, editors, *ICML*, volume 119, pages 89–96, Bonn, Germany, August 2005. ACM Press.

[297] W. Burkhard and R. Keller. Some approaches to best-match file searching. *Communications of the ACM*, 16(4):230–236, Apr. 1973.

[298] F. Burkowski. An algebra for hierarchically organized text-dominated databases. *Information Processing & Management*, 28(3):333–348, 1992.

[299] F. Burkowski. Retrieval activities in a database consisting of heterogeneous collections of structured text. In *15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Copenhagen, Denmark*, pages 112–125, 1992.

[300] A. W. Burks, H. H. Goldstine, and J. von Neumann. Preliminary discussion of the logical design of an electronic computing instrument. In W. Aspray and A. Burks, editors, *Papers of John von Neumann on Computers and Computer Theory*, pages 97–142. The MIT Press, Cambridge, MA, 1987. (originally appeared in 1946).

[301] M. Burner. Crawling towards eternity - building an archive of the world wide web. *Web Techniques*, 2(5), May 1997.

[302] M. Burrows and D. Wheeler. A block sorting lossless data compression algorithm. Technical Report 124, Digital Equipment Corporation, 1994.

[303] V. Bush. As we may think. *The Atlantic Monthly*, July 1945.

[304] S. Büttcher, C. L. Clarke, and G. V. Cormack. *Information Retrieval: Implementing and Evaluating Search Engines.* MIT Press, 2010.

[305] D. Byrd. A Scrollbar-based Visualization for Document Navigation. *Proceedings of the Fourth ACM International Conference on Digital Libraries*, 1999.

[306] D. Byrd and R. Podorozhny. Adding Boolean-quality control to best-match searching via an improved user interface. Technical Report IR-210, Computer Science Dept., Univ. of Massachusetts/Amherst, 2000.

[307] J. Byrum, Jr. Recommendations for urgently needed improvement of OPAC and the role of the national bibliographic agency in achieving it. In *World Library and Information Congress: 71th IFLA General Conference and Council, August 14-18, Oslo, Norway*, 2005. http://www.ifla.org/IV/ifla71/papers/124e-Byrum.pdf.

[308] F. Cacheda, V. Carneiro, V. Plachouras, and I. Ounis. Performance analysis of distributed information retrieval architectures using an improved network simulation model. *Information Processing and Management*, 43(1):204–224, 2007.

[309] B. Cahoon and K. McKinley. Performance evaluation of a distributed architecture for information retrieval. In *Proc. 19th Inter. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 110–118, Zurich, Switzerland, Aug. 1996.

[310] N. Caidi and A. Clement. Digital libraries and community networking: the canadian experience. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, page 386, Tucson, Arizona, 2004.

[311] P. Calado, B. A. Ribeiro-Neto, N. Ziviani, E. Moura, and I. Silva. Local versus global link information in the web. *ACM Trans. Inf. Syst.*, 21(1):42–63, January 2003.

[312] P. Calado, M. Cristo, E. S. de Moura, N. Ziviani, B. A. Ribeiro-Neto, and M. A. Gonçalves. Combining link-based and content-based methods for Web document classification. In *CIKM*, pages 394–401, 2003.

[313] P. Calado, M. Cristo, M. A. Gonçalves, E. S. de Moura, B. A. Ribeiro-Neto, and N. Ziviani. Link-based similarity measures for the classification of Web documents. *J. Am. Soc. Inf. Sci. Technol.*, 57(2):208–221, 2006.

[314] P. Calado, A. S. da Silva, A. H. F. Laender, B. A. Ribeiro-Neto, and R. C. Vieira. A Bayesian network approach to searching Web databases through keyword-based queries. *Inf. Process. Manage.*, 40(5):773–790, 2004.

[315] P. Calado, A. S. da Silva, R. C. Vieira, A. H. F. Laender, and B. A. Ribeiro-Neto. Searching Web databases by structuring keyword-based queries. In *CIKM*, pages 26–33, 2002.

[316] P. Calado, M. A. Gonçalves, E. A. Fox, B. A. Ribeiro-Neto, A. H. F. Laender, A. S. da Silva, D. C. Reis, P. A. Roberto, M. V. Vieira, and J. P. Lage. The web-DL environment for building digital libraries from the web. In *JCDL'03: Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 346–357, Houston, Texas, 2003.

[317] P. Calado and B. A. Ribeiro-Neto. An information retrieval approach for approximate queries. *IEEE Trans. Knowl. Data Eng.*, 15(1):236–239, 2003.

[318] J. Callan. Passage-level evidence in document retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 302–310. Springer-Verlag New York, Inc., 1994.

[319] J. Callan. Document filtering with inference networks. In *Proceedings of the 19th ACM SIGIR Conference*, pages 262–269, Zurich, Switzerland, August 1996.

[320] J. Callan. Distributed information retrieval. In W. B. Croft, editor, *Advances in Information Retrieval. Recent Research from the Center for Intelligent Information Retrieval*, volume 7 of *The Kluwer International Series on Information Retrieval*, chapter 5, pages 127–150. Kluwer Academic Publishers, Boston/Dordrecht/London, 2000.

[321] J. Callan, M. Connell, and A. Du. Automatic discovery of language models for text databases. In *Proceedings of ACM SIGMOD'99*, pages 479–490, New York, 1999.

[322] J. P. Callan, W. B. Croft, and S. M. Harding. The INQUERY retrieval system. In *DEXA*, pages 78–83, 1992.

[323] J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In E. A. Fox, P. Ingwersen, and R. Fidel, editors, *Proceedings of ACM SIGIR'95*, pages 21–28, Seattle, WA, July 1995. ACM Press.

[324] D. S. Callaway, M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Network robustness and fragility: Percolation on random graphs. *Phys. Rev. Lett.*, 85(25):5468–5471, Dec 2000.

[325] B. B. Cambazoglu, V. Plachouras, and R. Baeza-Yates. Quantifying performance and quality gains in distributed Web search engines. In J. Allan, J. A. Aslam, M. Sanderson, C. Zhai, and J. Zobel, editors, *SIGIR*, pages 411–418, Boston, MA, USA, July 2009. ACM.

[326] L. J. Camp. DRM: doesn't really mean digital copyright management. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 78–87, Washington, DC, USA, 2002.

[327] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, 1986.

[328] P. Cao and S. Irani. Cost-aware WWW proxy caching algorithms. In *USITS*, 1997.

[329] P. Cao and Z. Wang. Efficient top-K Query Calculation in Distributed Networks. In *PODS'04: Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART symposium on Principles of Database Systems*, Paris, France, 2004.

[330] Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon. Adapting ranking svm to document retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 186–193, Seattle, Washington, USA, 2006. ACM Press.

[331] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 129–136, Corvalis, Oregon, 2007. ACM Press.

[332] J. Carbonell and J. Goldstein. The use of MMR, Diversity-based reranking

for reordering documents and producing summaries. In *Proceedings of ACM SIGIR'98*, pages 335–336, Melbourne, Australia, 1998.

[333] D. Carmel, Y. Maarek, M. Mandelbrod, Y.Mass, and A. Soffer. Searching XML documents via XML fragments. In *26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Toronto, Canada*, pages 151–158, 2003.

[334] D. Carmel, Y. S. Maarek, and A. Soffer. XML and Information Retrieval: a SIGIR 2000 Workshop. *SIGIR Forum*, 34(1):31–36, 2000.

[335] D. Carmel and E. Yom-Tov. *Estimating the Query Difficulty for Information Retrieval.* Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan Claypool, 2010.

[336] D. Carmel, E. Yom-Tov, A. Darlow, and D. Pelleg. What makes a query difficult? In *Proc. of the ACM Int'l Conf. on Information Retrieval*, pages 390–397, 2006.

[337] Carnegie and Reuters. Reuters-21578 text categorization collection, 1987. Produced by Carnegie Group Inc. and Reuters Ltd., Reuters-21578 is made available for research purposes only. Data formatting and organization done by David Lewis.

[338] J. S. Carriére and R. Kazman. Webquery: searching and visualizing the Web through connectivity. *Computer Networks and ISDN Systems*, 29(8-13):1257–1267, September 1997.

[339] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *ICML '06: Proceedings of the 23rd International Conference on Machine learning*, pages 161–168, New York, NY, USA, 2006. ACM.

[340] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668–696, April 2008.

[341] Cassandra. http://incubator.apache.org/cassandra/, 2008.

[342] C. Castillo. *Effective Web Crawling.* PhD thesis, University of Chile, 2004.

[343] C. Castillo and R. Baeza-Yates. A new crawling model. In *Poster proceedings of the eleventh conference on World Wide Web*, Honolulu, Hawaii, USA, 2002.

[344] L. D. Catledge and J. E. Pitkow. Characterizing browsing strategies in the world-wide web. *Computer Networks and ISDN Systems*, 27(6):1065–1073, 1995.

[345] R. G. G. Cattell and D. K. Barry. *The Object Data Standard: ODMG 3.0.* Morgan Kaufmann, 2000.

[346] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, US, 1994.

[347] D. Chakrabarti, R. Kumar, and K. Punera. Quicklink selection for navigational query results. In *Proceedings of the Eighteenth International World Wide Web Conference*, Madrid, Spain, May 2009.

[348] S. Chakrabarti. Recent results in automatic Web resource discovery. *ACM Computing Surveys*, 31(4):17, 1999.

[349] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data.* Morgan Kaufmann, August 2002.

[350] S. Chakrabarti. Learning to rank in vector spaces and social networks. *Internet Mathematics*, 4(2-3):267–298, 2007.

[351] S. Chakrabarti, B. Dom, D. Gibson, S. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Experiments in topic distillation. In *ACM-SIGIR'98 Post Conference Workshop on Hypertext Information Retrieval for the Web*, Melbourne, Australia, 1998.

[352] S. Chakrabarti, B. Dom, P. Raghavan, S. Rajagopalan, D. Gibson, and J. Kleinberg. Automatic resource compilation by analyzing hyperlink structure and associated text. In *7th WWW Conference*, pages 65–74, Brisbane, Australia, April 1998.

[353] S. Chakrabarti, M. van den Berg, and B. Dom. Focused crawling: a new approach to topic-specific Web resource discovery. *Computer Networks*, 31(11–16):1623–1640, 1999.

[354] D. Chamberlin, J. Robie, and D. Florescu. Quilt: An XML Query Language for Heterogeneous Data Sources. In *The World Wide Web and Databases, Third International Workshop WebDB 2000, Dallas, Texas, USA, Selected Papers*, pages 1–25, 2000.

[355] C.-C. Chang and C.-J. Lin. LibSVM – A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

[356] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A distributed storage system for structured data. In *OSDI 2006*, pages 205–218, 2006.

[357] K. C. Chang, B. He, C. Li, M. Patel, and Z. Zhang. Structured databases on the web: observations and implications. *SIGMOD Rec.*, 33(3):61–70, September 2004.

[358] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning.* MIT Press, Cambridge, MA, 2006.

[359] E. Chávez, K. Figueroa, and G. Navarro. Effective proximity retrieval by ordering permutations. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(9):1647–1658, 2008.

[360] E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín. Proximity searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, Sept. 2001.

[361] C. Chen. Structuring and visualizing the WWW by generalized similarity analysis. In *8th ACM Conference on Hypertext and Hypermedia*, pages 177–186, Southampton, England, 1997.

[362] H. Chen, H. Jin, J. Wang, L. Chen, Y. Liu, and L. M. Ni. Efficient multi-keyword search over P2P Web. In *WWW'08: Proceeding of the 17th International World Wide Web conference*, Beijing, China, 2008.

[363] M. Chen, M. Hearst, J. Hong, and J. Lin. Cha-Cha: A System for Organizing Intranet Search Results. *Proceedings of the 2nd conference on USENIX Symposium on Internet Technologies and Systems*, pages 11–14, 1999.

[364] M. Chen, A. S. LaPaugh, and J. P. Singh. Predicting category accesses for a user in a structured information space. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 65–72, Tampere, Finland, July 2002. ACM.

[365] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Paterson. RAID: High-performance, reliable secondary storage. *ACM Comput. Surv.*, 26(2):145–185, June 1994.

[366] S. Chen and J. Goodman. *An Empirical Study of Smoothing Techniques for Language Modeling.* Harvard University, 1998. Tech Report TR-10-98.

[367] J. Cheney. Compressing XML with multiplexed hierarchical PPM models. In *Proc. 11th IEEE Data Compression Conference (DCC'01)*, pages 163–172, 2001.

[368] F. Cheong. *Internet agents spiders, wanderers, brokers and bots.* New Riders, 1996.

[369] E. H. Chi, P. Pirolli, K. Chen, and J. Pitkow. Using information scent to model user information needs and actions and the web. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 490–497, Seattle, WA, USA, 2001. ACM.

[370] E. H. Chi, P. Pirolli, and J. Pitkow. The scent of a site: a system for analyzing and predicting information scent, usage, and usability of a web site. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 161–168, The Hague, The Netherlands, 2000. ACM.

[371] Y. Chiaramella, P. Mulhem, and F. Fourel. A model for multimedia information retrieval. Technical report, University of Glasgow, 1996.

[372] T. T. Chinenyanga and N. Kushmerick. Expressive Retrieval from XML Documents. In *24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, Louisiana*, pages 163–171, 2001.

[373] J. Cho. The evolution of the Web and implications for an incremental crawler. In *Proceedings of 26th International Conference on Very Large Databases (VLDB)*, pages 527–534, Cairo, Egypt, September 2000. Morgan Kaufmann Publishers.

[374] J. Cho and H. Garcia-Molina. Synchronizing a database to improve freshness. In *Proceedings of ACM International Conference on Management of Data (SIGMOD)*, pages 117–128, Dallas, Texas, USA, 2000.

[375] J. Cho and H. Garcia-Molina. Parallel crawlers. In *Proceedings of the eleventh international conference on World Wide Web*, pages 124–135, Honolulu, Hawaii, USA, 2002. ACM Press.

[376] J. Cho and H. Garcia-Molina. Effective page refresh policies for Web crawlers. *ACM Transactions on Database Systems*, 28(4), 2003.

[377] J. Cho and H. Garcia-Molina. Estimating frequency of change. *ACM Transactions on Internet Technology*, 3(3), 2003.

[378] J. Cho, H. Garcia-Molina, and L. Page. Efficient crawling through URL ordering. In *Proceedings of the seventh conference on World Wide Web*, Brisbane, Australia, 1998. Elsevier Science.

[379] J. Cho, N. Shivakumar, and H. Garcia-Molina. Finding replicated Web collections. In *ACM SIGMOD*, pages 355–366, 1999.

[380] A. Chowdhury, O. Frieder, D. A. Grossman, and M. C. McCabe. Collection statistics for fast duplicate document detection. *ACM Trans. Inf. Syst.*, 20(2):171-191, 2002.

[381] A. Chowdhury and G. Pass. Operational requirements for scalable search systems. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 435–442, New York, NY, USA, 2003. ACM Press.

[382] G. Chowdhury. *Introduction to Modern Information Retrieval.* Facet Publishing, 2003. 488 pages.

[383] M. G. Christel, D. B. Winkler, and C. R. Taylor. Multimedia abstractions for a digital video library. In *DL '97: Proceedings of the Second ACM International Conference on Digital libraries*, pages 21–29, New York, NY, USA, 1997. ACM.

[384] K. Church and W. Gale. Poisson mixtures. *Natural Language Engineering*, 1(2):163–190, 1995.

[385] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proc. of VLDB Conf.*, pages 426–435, Athens, Greece, Aug. 1997.

[386] Citeseer. http://citeseer.ist.psu.edu/, 1997.

[387] CiteseerX. http://citeseerX.ist.psu.edu/, 2008.

[388] CITIDEL. Computing and Information Technology Interactive Digital Educational Library. http://www.citidel.org, 2004.

[389] C. Clarke. Controlling overlap in content-oriented XML retrieval. In *28th annual international ACM SIGIR conference on Research and development in information retrieval, Salvador, Brazil*, pages 314–321, 2005.

[390] C. Clarke, E. Agichtein, S. Dumais, and R. White. The influence of caption features on clickthrough patterns in Web search. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'07)*, pages 135–142. ACM Press New York, NY, USA, 2007.

[391] C. Clarke, G. Cormack, and F. Burkowski. Shortest substring ranking (multitext experiments for TREC-4). In D. Harman, editor, *Proceedings of the Fifth Text REtrieval Conference (TREC-5)*, 1996.

[392] C. L. Clarke, G. V. Cormack, and F. J. Burkowski. An algebra for structured text search and a framework for its implementation. *The Computer Journal*, 38:43–56, 1995.

[393] CLEF 2009. http://clef-campaign.org/2009/2009_agenda.html, 2009.

[394] C. Cleverdon. Report on the testing and analysis of an investigation into the comparative efficiency of indexing systems. Technical report, College of Aeronautics, Cranfield, 1962.

[395] C. Cleverdon. The Cranfield tests on index language devices. *ASLIB Proceedings*, 19(6):173–194, 1967.

[396] C. Cleverdon. Optimizing convenient online access to bibliographic databases. In *Document retrieval systems*, pages 32–41. Taylor Graham Publishing, London, UK, UK, 1988.

[397] C. Cleverdon, J. Mills, and M. Keen. Factors determining the performance of indexing systems. Technical report, ASLIB, 1966.

[398] C. Cleverdon and R. Thorne. An experiment with the uniterm system. Technical Report Library Memo 7, RAE, 1954.

[399] C. W. Cleverdon. The significance of the Cranfield tests on index languages. In *SIGIR '91: Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12, 1991.

[400] CliffsNotes. About pride and prejudice — publication history and critical reception, 2009. http://www.cliffsnotes.com/WileyCDA/LitNote/Pride-and-Prejudice-About-Pride-and-Prejudice-Publication-History-and-Critical-Reception.id-147,pageNum-9.html.

[401] CMU. WebKB hypertext collection. http://www.cs.cmu.edu/~webkb/.

[402] CMU. 20 newsgroup, 1999. Originally owned by Tom Mitchell, Computer Science Department, Carnegie Mellon University.

[403] T. A. S. Coelho, P. P. Calado, L. V. Souza, B. A. Ribeiro-Neto, and R. Muntz. Image retrieval using multiple evidence ranking. *IEEE Transactions on Knowledge and Data Engineering*, 16(4):408–417, 2004.

[404] E. G. Coffman, Z. Liu, and R. R. Weber. Optimal robot scheduling for Web search engines. *Journal of Scheduling*, 1(1):15–29, 1998.

[405] S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv. XSEarch: A Semantic Search Engine for XML. In *29th International Conference on Very Large Data Bases, Berlin, Germany*, pages 45–56, 2003.

[406] W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. *ACM Transaction on Office and Information Systems*, 17(2):141–173, 1999.

[407] W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.

[408] D. E. Comer and D. L. Stevens. *Internetworking with TCP/IP Vol III: Client-Server Programming and Applications*. Prentice-Hall, Inc., Englewood Cliffs, NJ, USA, 1993.

[409] B. Commentz-Walter. A string matching algorithm fast on the average. In *Proc. ICALP'79*, pages 118–132. Springer-Verlag, 1979.

[410] Communications of the ACM. Hypermedia. February 1994. 37(2).

[411] Communications of the ACM. Hypermedia, August 1995. 38(8).

[412] J. Conklin. Hypertext: An introduction and survey. *IEEE Computer*, 20(9):17–41, Sept. 1987.

[413] D. Connolly. Dan Connolly on the Architecture of the Web: Let a Thousand Flowers Bloom. *IEEE Internet Computing*, 2(2):22–31, 1998.

[414] N. E. O. Connor, S. Marlow, N. Murphy, A. Smeaton, P. Browne, S. Deasy, H. Lee, and K. McDonald. Fischlar: An on-line system for indexing and broadcasting television content. *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, 3:1633 – 1636, 2001.

[415] M. Consens and T. Milo. Algebras for Querying Text Regions. In *Proceedings of the Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, San Jose, California*, pages 11–22, 1995.

[416] M. P. Consens and A. O. Mendelzon. The G+/GraphLog visual query system. In H. Garcia-Molina and H. V. Jagadish, editors, *SIGMOD Conference*, page 388, Atlantic City, NJ, USA, May 1990. ACM Press.

[417] Consultative Committee for Space Data Systems. Reference model for an open archival information system (OAIS), 2001. http://public.ccsds.org/publications/archive/650x0b1.pdf.

[418] M. Cooke and D. Ellis. The auditory organization of speech and other sources in listeners and computational models. *Speech Communications*, pages 141–177, 2001.

[419] R. Cooley, B. Mobasher, and J. Srivastava. Web mining: Information and pattern discovery on the world wide web. In *ICTAI*, pages 558–567, 1997.

[420] A. Cooper. A survey of query log privacy-enhancing techniques from a policy perspective. *ACM Transactions on the Web (TWeb)*, 2(4), 2008.

[421] B. F. Cooper and H. Garcia-Molina. Sil: a model for analyzing scalable peer-to-peer search networks. *Comput. Netw.*, 50(13):2380-2400, 2006.

[422] W. S. Cooper. The formalism of probability theory in IR: A foundation or an encumbrance? In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Probabilistic Models, pages 242–247, 1994. Triennial ACM-SIGIR Award Paper.

[423] W. S. Cooper, F. C. Gey, and D. P. Dabney. Probabilistic retrieval based on staged logistic regression. In *Proc. of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 198–210, Copenhagen, Denmark, 1992.

[424] J. Cope, N. Craswell, and D. Hawking. Automatic discovery of search interfaces on the web. In *The Fourteenth Australasian Database Conference*, volume 17 of *Conferences in Research and Practice in Information Technology*, Adelaide,

Australia, 2003.

[425] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press/McGraw-Hill, Cambridge, MA, 1990.

[426] D. E. Corporation. AltaVista, 1996. `http://altavista.com`.

[427] Corporation for National Research Initiatives, the Handle System, May 1998. http://www.handle.net/.

[428] P. Correia-Saraiva, E. Silva de Moura, N. Ziviani, W. Meira, R. Fonseca, and B. A. Ribeiro-Neto. Rank-preserving two-level caching for scalable search engines. In *Proceedings of the 24th International ACM Conference on Research and Development in Information Retrieval*, September 2001. New Orleans, USA.

[429] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[430] E. Cortez, A. S. da Silva, M. A. Gonçalves, F. Mesquita, and E. S. de Moura. FLUX-CM: Flexible Unsupervised Extraction of Citation Metadata. In *JCDL '07: Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, 2007.

[431] I. G. Councill, C. L. Giles, H. Han, and E. Manavoglu. Automatic acknowledgement indexing: expanding the semantics of contribution in the citeseer digital library. In *Proceedings of the 3rd International Conference on Knowledge Capture (K-CAP 2005)*, pages 19–26, Banff, Alberta, Canada, 2005.

[432] T. Couto, M. Cristo, M. A. Gonçalves, P. Calado, N. Ziviani, E. Moura, and B. A. Ribeiro-Neto. A comparative study of citations and links in document classification. In *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 75–84, New York, NY, USA, 2006. ACM Press.

[433] M. Covell and S. Ahmad. Analysis-by-synthesis dissolve detection. In *Proceedings of IEEE International Conference on Image Processing*, Rochester, NY, September 2002.

[434] M. Covell, M. Withgott, and M. Slaney. Mach1: Nonuniform time-scale modification of speech. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Language Processing*, volume 1, pages 349–352, 1998.

[435] I. J. Cox, M. L. Miller, S. M. Omohundro, and P. N. Yianilos. Pichunter: Bayesian relevance feedback for image retrieval. *International Conference on Pattern Recognition*, 13:361–369, 1996.

[436] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

[437] N. Craswell, P. Bailey, and D. Hawking. Server selection on the world wide web. In *Proceedings of the 5th ACM Digital Libraries Conference*, pages 37–46, June 2000.

[438] N. Craswell, F. Crimmins, D. Hawking, and A. Moffat. Performance and cost tradeoffs in Web search. In *Proceedings of the 15th Australasian Database Conference*, pages 161–169, Dunedin, New Zealand, January 2004.

[439] N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec 2005 enterprise track. In *The Fourteenth Text REtrieval Conference (TREC 2005) Proceedings*, Gaithersburg, MD, 2005. NIST. TREC Special Publication: SP 500-266. trec.nist.gov/pubs/trec14/papers/ENTERPRISE.OVERVIEW.pdf.

[440] N. Craswell, D. Hawking, A.-M. Vercoustre, and P. Wilkins. P@noptic expert: Searching for experts not just for documents. In *Poster Proceedings of AusWeb'01*, 2001. /urlausweb.scu.edu.au/aw01/papers/edited/vercoustre/paper.htm.

[441] A. Crauser and P. Ferragina. A theoretical and experimental study on the construction of suffix arrays in external memory. *Algorithmica*, 32(1):1–35, 2002.

[442] A. Crespo and H. Garcia-Molina. Archival storage for digital libraries. In *DL'98: Proceedings of the 3rd ACM International Conference on Digital Libraries*, pages 69–78, 1998.

[443] A. Crespo and H. Garcia-Molina. Semantic overlay networks for P2P systems. Technical Report 2003-75, Stanford University, 2003.

[444] F. Crestani, M. Lalmas, C. J. van Rijsbergen, and I. Campbell. "Is this document relevant? ... probably": A survey of probabilistic models in information retrieval. *ACM Computing Surveys*, 30(4):528–552, Dec. 1998.

[445] M. Cristo, P. Calado, M. de Lourdes da Silveira, I. Silva, R. R. Muntz, and B. A. Ribeiro-Neto. Bayesian belief networks for ir. *Int. J. Approx. Reasoning*, 34(2-3):163–179, 2003.

[446] M. Cristo, P. Calado, E. S. de Moura, N. Ziviani, and B. A. Ribeiro-Neto. Link information as a similarity measure in Web classification. In *SPIRE*, pages 43–55, 2003.

[447] M. Crochemore and W. Rytter. *Text Algorithms.* Oxford University Press, Oxford, UK, 1994.

[448] M. Crochemore and W. Rytter. *Jewels of Stringology — Text algorithms.* World Scientific, 2002. ISBN 981-02-4782-6. 320 pages.

[449] B. Croft, D. Metzler, and T. Strohman. *Search Engines - Information Retrieval in Practice.* Addison Wesley, February 2009.

[450] W. Croft. Experiments with representation in a document retrieval system. *Information Technology: Research and Development*, 2(1):1–21, 1983.

[451] W. Croft, S. Cronen-Townsend, and V. Lavrenko. Relevance Feedback and Personalization: A Language Modeling Perspective. *Delos Workshop: Personalisation and Recommender Systems in Digital Libraries*, 2001.

[452] W. Croft and D. Harper. Using probabilistic models of retrieval without relevance information. *Journal of Documentation*, 35(4):285–295, 1979.

[453] W. B. Croft and J. Lafferty, editors. *Language Modeling for Information Retrieval.* Number 13 in the Information Retrieval Series. Kluwer/Springer, 2003.

[454] S. Cronen-Townsend, Y. Zhou, and W. Croft. A Language Modeling Framework for Selective Query Expansion. Technical Report Technical Report IR-338, Center for Intelligent Information Retrieval, University of Massachusetts, 2004.

[455] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306, 2002.

[456] T. Crook, B. Frasca, R. Kohavi, and R. Longbotham. Seven pitfalls to avoid when running controlled experiments on the web. In J. F. E. IV, F. Fogelman-Soulié, P. A. Flach, and M. J. Zaki, editors, *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, pages 1105–1114. ACM, 2009.

[457] C. J. Crouch and B. Yang. Experiments in automatic statistical thesaurus construction. In *Proc. ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 77–88, Denmark, 1992.

[458] M. E. Crovella and A. Bestavros. Self-similarity in world wide Web traffic: evidence and possible causes. In *SIGMETRICS '96: Proceedings of the 1996 ACM SIGMETRICS international conference on Measurement and modeling*

of computer systems, volume 24, pages 160–169, New York, NY, USA, May 1996. ACM Press.

[459] R. Crow. The case for institutional repositories: A SPARC position paper. Technical report, The Scholarly Publishing & Academic Resources Coalition, Washington, D.C., Aug. 2002.

[460] Computer Science Bibliography. http://liinnwww.ira.uka.de/bibliography.

[461] S. Cucerzan and E. Brill. Spelling correction as an iterative process that exploits the collective knowledge of Web users. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 293–300, Barcelona, Spain, July 2004.

[462] F. M. Cuenca-Acuna and T. D. Nguyen. Text-based content search and retrieval in ad-hoc P2P communities. In *Revised Papers from the NETWORKING 2002 Workshops on Web Engineering and Peer-to-Peer Computing*, pages 220–234, London, UK, 2002. Springer-Verlag.

[463] J. Culpepper and A. Moffat. Compact set representation for information retrieval. In *SPIRE 2007*, volume 4726 of *Lecture Notes in Computer Science*, pages 137-148. Springer, 2007.

[464] S. J. Cunningham, N. Reeves, and M. Britland. An ethnographic study of music information seeking: implications for the design of a music digital library. In *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries*, Portland, Oregon, 2003.

[465] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana. Unraveling the Web services web: An introduction to SOAP, WSDL, and UDDI. *IEEE Distributed Systems Online*, 3(4), 2002.

[466] E. Cutrell, D. Robbins, S. Dumais, and R. Sarin. Fast, flexible filtering with Phlat. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'06)*, pages 261–270, 2006.

[467] D. Cutting, J. Pedersen, D. Karger, and J. Tukey. Scatter/Gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'92)*, pages 318-329, Copenhagen, Denmark, 1992.

[468] M. Czerwinski, M. van Dantzich, G. Robertson, and H. Hoffman. The contribution of thumbnail image, mouse-over text and spatial location memory to Web page retrieval in 3D. *Proceedings of Human-Computer Interaction (INTERACT'99)*, 99:163–170, 1999.

[469] A. Czumaj, M. Crochemore, L. Gasieniec, S. Jarominck, T. Lecroq, W. Plandowski, and W. Rytter. Speeding up two string-matching algorithms. *Algorithmica*, 12:247–267, 1994.

[470] L. C. da Rocha, F. Mourão, A. Pereira, M. A. Gonçalves, and W. Meira Jr. Exploiting temporal contexts in text classification. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, Napa, California, USA, October 26-30, 2008*, pages 243–252, 2008.

[471] A. S. da Silva, P. Calado, R. C. Vieira, A. H. F. Laender, and B. A. Ribeiro-Neto. Keyword-based queries over Web databases. In *Effective Databases for Text & Document Management*, pages 74–92. 2003.

[472] A. S. da Silva, E. A. Veloso, P. B. Golgher, A. H. F. Laender, and N. Ziviani. CoBWeb - a crawler for the Brazilian web. In *Proceedings of String Processing and Information Retrieval (SPIRE)*, pages 184–191, Cancun, México, 1999. IEEE CS Press.

[473] R. da Silva Torres, C. B. Medeiros, M. A. Gonçalves, and E. A. Fox. A digital library framework for biodiversity information systems. *International Journal on Digital Libraries*, 6(1):3–17, 2006.

[474] R. Darnell. *HTML 4.0 Unleashed, Professional Reference Edition*. Samms.net Publishing, 1998.

[475] J. R. Davis and C. Lagoze. NCSTRL: Design and deployment of a globally distributed digital library. *Journal of the American Society for Information Science*, 51(3):273–280, 2000.

[476] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28:357–366, August 1980.

[477] B. D. Davison. Topical locality in the web. In *Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval*, pages 272-279, Athens, Greece, 2000. ACM Press.

[478] Digital Bibliography & Library Project. http://www.informatik.uni-trier.de/~ley/db/.

[479] H. M. de Almeida, M. A. Gonçalves, M. Cristo, and P. Calado. A combined component approach for finding collection-adapted ranking functions based on genetic programming. In *SIGIR 2007: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, July 23-27, 2007*, pages 399–406. ACM, 2007.

[480] R. de Freitas Vale, B. A. Ribeiro-Neto, L. R. S. de Lima, A. H. F. Laender, and H. R. Freitas-Junior. Improving text retrieval in medical collections through automatic categorization. In *SPIRE*, pages 197-210, 2003.

[481] M. de Lourdes da Silveira and B. A. Ribeiro-Neto. Concept-based ranking: a case study in the juridical domain. *Inf. Process. Manage.*, 40(5):791–805, 2004.

[482] M. de Lourdes da Silveira, B. A. Ribeiro-Neto, R. de Freitas Vale, and R. T. Assumpção. Vertical searching in juridical digital libraries. In *ECIR*, pages 491–501, 2003.

[483] A. de Vries, G. Kazai, and M. Lalmas. Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *RIAO 2004 Conference on Coupling approaches, coupling media and coupling languages for information retrieval, Vaucluse, France*, pages 463–473, 2004.

[484] J. Dean. Challenges in building large-scale information retrieval systems: invited talk presentation. In R. Baeza-Yates, P. Boldi, B. A. Ribeiro-Neto, and B. B. Cambazoglu, editors, *WSDM*, page 1, Barcelona, Spain, 2009. ACM.

[485] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of ACM*, 51(1):107 113, 2008. Preliminary version published in OSDI 2004, p. 137-150.

[486] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: Amazon's highly available key-value store. *SIGOPS Oper. Syst. Rev.*, 41(6):205-220, 2007.

[487] K. J. Delaney. Yahoo Contends It Tops Google in Number of Pages Searched. Wall Street Journal, August 15, 2005, page B4.

[488] A. Delgado and R. Baeza-Yates. An analysis of query languages for XML. *UPGRADE. The European Online Magazine for the IT Professional, Special Issue on Information Retrieval and the Web*, III(3), 2002.

[489] DELOS Network of Excellence. Reference Model for Digital Library Management Systems. http://www.delos.info/ReferenceModel.

[490] E. D. Demaine, A. López-Ortiz, and J. I. Munro. Adaptive set intersections, unions, and differences. In *SODA*, pages 743–752, San Francisco, CA, USA, 2000.

[491] E. D. Demaine, A. López-Ortiz, and J. I. Munro. Experiments on adaptive set intersections for text retrieval systems. In A. L. Buchsbaum and J. Snoeyink, editors, *ALENEX*, volume 2153 of *Lecture Notes in Computer Science*, pages 91–104, Washington, DC, USA, January 2001. Springer.

[492] S. Dennis, R. Mcarthur, and P. Bruza. Searching the World Wide Web Made Easy? the Cognitive Load Imposed By Query Refinement Mechanisms. *Proceedings of the Australian Document Computing Conference*, pages 65–71, 1998.

[493] L. Denoyer and P. Gallinari. The Wikipedia XML Corpus. *SIGIR Forum*, 40(1):64–69, 2006.

[494] A. Deutsch, M. Fernández, D. Florescu, A. Levy, and D. Suciu. XML-QL. In *Query Languages 1998*, 1998.

[495] Z. Dezso, E. Almaas, A. Lukacs, B. Racz, I. Szakadat, and A. Barabasi. Fifteen minutes of fame: the dynamics of information access on the web, May 2005.

[496] F. Diaz and R. Jones. Using temporal profiles of queries for precision prediction. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 18–24, 2004.

[497] M. Diligenti, F. Coetzee, S. Lawrence, L. C. Giles, and M. Gori. Focused crawling using context graphs. In *Proceedings of 26th International Conference on Very Large Databases (VLDB)*, pages 527–534, Cairo, Egypt, September 2000.

[498] S. Dill, R. Kumar, K. S. Mccurley, S. Rajagopalan, D. Sivakumar, and A. Tomkins. Self-similarity in the web. *ACM Trans. Inter. Tech.*, 2(3):205 223, 2002.

[499] J. Dinet, M. Favart, and J. Passerault. Searching for information in an online public access catalogue(OPAC): the impacts of information search expertise on the use of Boolean operators. *Journal of Computer Assisted Learning*, 20(5):338-346, 2004.

[500] S. Ding, J. He, H. Yan, and T. Suel. Using graphics processors for high performance IR query processing. In J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, editors, *WWW*, pages 421–430, Madrid, Spain, 2009. ACM.

[501] A. Divoli, M. Hearst, and M. Wooldridge. Evidence for showing gene/protein name suggestions in bioscience literature search interfaces. *Pacific Symposium on Biocomputing*, 568:79, 2008.

[502] P. A. Dmitriev, N. Eiron, M. Fontoura, and E. Shekita. Using annotations in enterprise search. In *Proceedings of WWW'06*, pages 811–817, New York, NY, USA, 2006. ACM.

[503] M. Dodge. The geography of cyberspace directory. http://personalpages.manchester.ac.uk/staff/m.dodge/cybergeography/geography_Of_Cyberspace.html, 1997-2004.

[504] P. Dömel. Webmap: A graphical hypertext navigation tool. In *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*, 1994. http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/WWW2_Proceedings.html.

[505] S. Dominich. *Mathematical foundations of information retrieval*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[506] D. Donato, S. Leonardi, S. Millozzi, and P. Tsaparas. Mining the inner structure of the Web graph. In *Eigth international workshop on the Web and databases WebDB*, Baltimore, USA, June 2005.

[507] P. Donmez, K. M. Svore, and C. J. Burges. On the local optimality of LambdaRank. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 460–467, Boston, MA, USA, 2009. ACM Press.

[508] Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *WWW'07: Proceedings of the 16th international conference on World Wide Web*, pages 581–590, New York, NY, USA, 2007. ACM.

[509] F. Douglis, A. Feldmann, B. Krishnamurthy, and J. C. Mogul. Rate of change and other metrics: a live study of the world wide web. In *USENIX Symposium on Internet Technologies and Systems*, pages 147–158, Monterey, California, USA, December 1997.

[510] C. Doulkeridis, K. Nørvåg, and M. Vazirgiannis. Peer-to-peer similarity search over widely distributed document collections. In *LSDS-IR '08: Proceeding of the 2008 ACM workshop on Large-Scale distributed systems for information retrieval*, pages 35–42, New York, NY, USA, 2008. ACM.

[511] D. Dreilinger. Savvy Search, 1996. http://savvy.cs.colostate.edu:2000/form?beta.

[512] I. Drost and T. Scheffer. Thwarting the nigritude ultramarine: learning to identify link spam. In *Proceedings of the 16th European Conference on Machine Learning (ECML)*, volume 3720 of *Lecture Notes in Artificial Intelligence*, pages 233–243, Porto, Portugal, 2005.

[513] D. D'Souza and J. A. Thom. Collection Selection Using n-Term Indexing. In *Proceedings of CODAS*, pages 52–63, 1999.

[514] D. D'Souza, J. A. Thom, and J. Zobel. Collection selection for managed distributed document databases. *Information Processing & Management*, 40, 2004.

[515] D. D'Souza, J. Zobel, and J. Thom. Is CORI effective for collection selection? an exploration of parameters, queries, and data. In *Proceedings of Australian Document Computing Symposium*, pages 41–46, Melbourne, Australia, 2004.

[516] M. Dubinko, R. Kumar, J. Magnani, J. Novak, P. Raghavan, and A. Tomkins. Visualizing tags over time. In *WWW'06: Proceedings of the 15th International Conference on World Wide Web*, pages 193–202, New York, NY, USA, 2006. ACM Press.

[517] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2000.

[518] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins. Stuff I've seen: a system for personal information retrieval and re-use. In *Proceedings of ACM SIGIR '03*, pages 72–79, Toronto, Canada, 2003. ACM.

[519] S. Dumais, E. Cutrell, and H. Chen. Optimizing search by showing results in context. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'01)*, pages 277–284, 2001.

[520] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 148–155, 1998.

[521] S. T. Dumais and H. Chen. Hierarchical classification of Web content. In *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, pages 256–263, Athens, GR, 2000.

[522] G. Dupret and B. Piwowarski. A user browsing model to predict search engine

click data from past observations. In S.-H. Myaeng, D. W. Oard, F. Sebastiani, T.-S. Chua, and M.-K. Leong, editors, *SIGIR*, pages 331–338. ACM, November 2008.

[523] S. Dziadosz and R. Chandrasekar. Do Thumbnail Previews Help Users Make Better Relevance Decisions about Web Search Results. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'02)*, pages 365–366, 2002.

[524] J.-P. Eckmann and E. Moses. Curvature of co-links uncovers hidden thematic layers in the World Wide Web. *PNAS*, 99(9):5825–5829, April 2002.

[525] J. Edwards, K. S. Mccurley, and J. A. Tomlin. An adaptive model for optimizing performance of an incremental Web crawler. In *Proceedings of the Tenth Conference on World Wide Web*, pages 106–113, Hong Kong, May 2001. Elsevier Science.

[526] R. Edwards, S. Clarke, and A. Kellett. Organisations waste 10% of salary bill searching for information. http://www.butlergroup.com/pdf/PressReleases/ESRReportPressRelease.pdf, October 2006.

[527] D. Egan, J. Remde, L. Gómez, T. Landauer, J. Eberhardt, and C. Lochbaum. Formative design evaluation of SuperBook. *Transaction on Information Systems*, 7(1), 1989.

[528] D. Egan, J. Remde, T. Landauer, C. Lochbaum, and L. Gómez. Behavioral evaluation and analysis of a hypertext browser. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'89)*, pages 205–210, May 1989.

[529] D. Eichmann. The RBSE spider: balancing effective search against Web load. In *Proceedings of the first World Wide Web Conference*, Geneva, Switzerland, May 1994.

[530] S. G. Eick and G. J. Wills. Navigating large networks with hierarchies. In *Proc. of the Conference on Visualization '93*, pages 204–209, San Jose, Oct. 1993.

[531] N. Eiron, K. S. Curley, and J. A. Tomlin. Ranking the Web frontier. In *Proceedings of the 13th international conference on World Wide Web*, pages 309–318, New York, NY, USA, 2004. ACM Press.

[532] P. Elias. Universal codeword sets and representations of the integers. *IEEE Transactions on Information Theory*, 21:194–203, 1975.

[533] E. Elliott and G. Davenport. Video streamer. *ACM CHI-94: Proceedings on Human Factors in Computing Systems: Celebrating Independence*, pages 65–66, 1994.

[534] Endeca. Endeca. http://www.endeca.com/.

[535] EPrints. Registry of open access repositories (ROAR), 2006. http://roar.eprints.org/.

[536] P. Erdõs and A. Rényi. Random graphs. *Publication of the Mathematical Institute of the Hungarian Acadamy of Science*, 5, 1960.

[537] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. W. Han, and U. Fayyad, editors, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231. AAAI Press, 1996.

[538] J. Exposto, J. Macedo, A. Pina, A. Alves, and J. Rufino. Geographical partition for distributed Web crawling. In *GIR '05: Proceedings of the 2005 workshop on Geographic information retrieval*, pages 55–60, Bremen, Germany, 2005. ACM Press.

[539] G. Eysenbach and C. Kohler. How do consumers search for and appraise health information on the world wide web? Qualitative study using focus groups, usability tests, and in-depth interviews. *British Medical Journal*, 324(7337):573-577, 2002.

[540] Facebook. http://www.facebook.com/, 2004.

[541] R. Fagin, R. Kumar, K. S. McCurley, J. Novak, D. Sivakumar, J. A. Tomlin, and D. P. Williamson. Searching the workplace web. In *Proceedings of WWW2003*, Budapest, Hungary, May 2003. http://www2003.org/cdrom/papers/refereed/p641/xhtml/p641-mccurley.html.

[542] R. Fagin, A. Lotem, and M. Naor. Optimal aggregation algorithms for middleware. In *PODS'01: Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART symposium on Principles of Database Systems*, Santa Barbara, CA, USA, 2001.

[543] R. Fagin and E. L. Wimmers. Incorporating user preferences in multimedia queries. In *Proceedings of the 1997 International Conference on Database Theory*, pages 247-261, 1997.

[544] T. Fagni, R. Perego, F. Silvestri, and S. Orlando. Boosting the performance of Web search engines: Caching and prefetching query results by exploiting historical usage data. *ACM Trans. Inf. Syst.*, 24(1):51-78, 2006.

[545] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and effective querying by image content. *J. of Intelligent Information Systems*, 3(3/4):231-262, July 1994.

[546] C. Faloutsos and R. Chan. Text access methods for optical and large magnetic disks: design and performance comparison. In *Proc. of VLDB'88*, pages 280-293, 1988.

[547] C. Faloutsos and S. Christodoulakis. Description and performance analysis of signature file methods. *ACM TOIS*, 5(3):237-257, 1987.

[548] C. Faloutsos and V. Gaede. Analysis of n-dimensional quadtrees using the Hausdorff fractal dimension. In *Proc. of VLDB Conf.*, pages 40-50, Bombay, India, Sept. 1996.

[549] C. Faloutsos and I. Kamel. Beyond uniformity and independence: Analysis of R-trees using the concept of fractal dimension. In *Proc. ACM SIGACT-SIGMOD-SIGART PODS*, pages 4-13, Minneapolis, MN, May 1994.

[550] A. Fariña. *New Compression Codes for Text Databases*. PhD thesis, Computer Science Department, University of A Coruña, A Coruña, Spain, 2005.

[551] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. In *IEEE CVPR 2004, Workshop on Generative-Model Based Vision*, 2004.

[552] R. Feldman and J. Sanger. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, Cambridge, MA, USA, December 2006.

[553] S. Feldman and C. Sherman. The high cost of not finding information. White Paper #29127, IDC, April 2003. http://www.idc.com.

[554] T. Ferl and L. Millsap. The knuckle-cracker's dilemma: a transaction log study of OPAC subject searching. *Information Technology and Libraries*, 15(2):113-126, 1996.

[555] M. Fernandez, D. Florescu, A. Levy, and D. Suciu. A query language for a Web-site management system. *SIGMOD Record*, 26(3):4-11, September 1997.

[556] P. Ferragina and G. Manzini. Opportunistic data structures with applications. In *Proc. 41st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 390-398, 2000.

[557] P. Ferragina and G. Manzini. Indexing compressed texts. *Journal of the ACM*, 52(4):552-581, 2005.

[558] P. Ferragina, G. Manzini, V. Mäkinen, and G. Navarro. Compressed representation of sequences and full-text indexes. *ACM Transactions on Algorithms*, 3(2), 2007. Earlier version in *Proc. SPIRE 2004*.

[559] D. Fetterly, M. Manasse, and M. Najork. On the evolution of clusters of near-duplicate Web pages. *Journal of Web Engineering*, 2(4):228 246, 2004.

[560] D. Fetterly, M. Manasse, and M. Najork. Detecting phrase-level duplication on the world wide web. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 170–177, New York, NY, USA, 2005. ACM Press.

[561] D. Fetterly, M. Manasse, M. Najork, and J. L. Wiener. A large-scale study of the evolution of Web pages. In *Proceedings of the Twelfth Conference on World Wide Web*, Budapest, Hungary, 2003. ACM Press.

[562] S. Few. *Information Dashboard Design: the Effective Visual Communication of Data*. O'Reilly, 2006.

[563] S. Few. *Now You See It: Simple Visualization Techniques for Quantitative Analysis*. Analytics Press, 2009.

[564] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. RFC 2616 - HTTP/1.1, the hypertext transfer protocol. http://w3.org/Protocols/rfc2616/rfc2616.html, 1999.

[565] R. A. Finkel, A. B. Zaslavsky, K. Monostori, and H. W. Schmidt. Signature extraction for overlap detection in documents. In M. J. Oudshoorn, editor, *Twenty-Fifth Australasian Computer Science Conference (ACSC2002)*, volume 4 of *CRPIT*, pages 59–64, Melbourne, Australia, 2002. Australian Computer Society.

[566] K. Fishkin and M. C. Stone. Enhanced dynamic queries via movable filters. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'95)*, volume 1, pages 415–420, 1995.

[567] L. Fitzpatrick and M. Dent. Automatic feedback using past queries: Social searching? *20th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 306–313, 1997.

[568] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: the QBIC System. *Computer*, 28(9):23–32, 1995.

[569] D. Florescu, A. Y. Levy, and A. O. Mendelzon. Database techniques for the world-wide web: A survey. *SIGMOD Record*, 27(3):59–74, 1998.

[570] M. J. Flynn. Very high-speed computing systems. In *Proc. IEEE*, volume 54, pages 1901–1909, 1966.

[571] B. M. Fonseca, P. B. Golgher, E. S. De Moura, and N. Ziviani. Using association rules to discovery search engines related queries. In *First Latin American Web Congress (LA-WEB'03)*, November, 2003. Santiago, Chile.

[572] B. M. Fonseca, P. B. Golgher, B. Pôssas, B. A. Ribeiro-Neto, and N. Ziviani. Concept-based interactive query expansion. In *CIKM*, pages 696–703, 2005.

[573] E. Forgy. Cluster analysis of multivariate data: Efficiency versus interpretability of classification. *Biometrics*, 21:768–780, 1965.

[574] D. Foskett. Thesaurus. In K. S. Jones and P. Willet, editors, *Readings in Information Retrieval*, pages 111–134. Morgan Kaufmann Publishers, Inc., 1997.

[575] M. Foulonneau, Cole, T. W., Habing, T. G., Shreeves, and S. L. Using collection descriptions to enhance an aggregation of harvested item-level metadata. In *JCDL'05: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 32–41, Denver, Colorado, USA, 2005.

[576] F. Fouss, M. Saerens, and J.-M. Renders. Links between kleinberg's hubs and authorities, correspondence analysis, and Markov chains. In *Proceedings of the third IEEE international conference on data mining (ICDM)*, pages 521–524, Melbourne, Florida, USA, November 2003.

[577] C. Fox. Lexical analysis and stoplists. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Data Structures & Algorithms*, pages 102–130. Prentice Hall, 1992.

[578] E. A. Fox, R. K. France, E. Sahle, A. Daoud, and B. E. Cline. Development of a modern OPAC: From REVTOLC to MARIAN. In *Proc. of the 16th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 248–259, 1993.

[579] E. A. Fox and S. Urs. Digital Libraries. In B. Cronin, editor, *Annual Review of Information Science and Technology*, volume 36, Ch. 12, pages 503–589. American Society for Information Science and Technology, 2002.

[580] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. Evaluating implicit measures to improve Web search. *ACM Trans. Inf. Syst.*, 23(2):147–168, 2005.

[581] W. Frakes. Stemming algorithms. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Data Structures & Algorithms*, pages 131–160. Prentice Hall, 1992.

[582] W. Frakes and R. Baeza-Yates, editors. *Information Retrieval: Data Structures & Algorithms*. Prentice Hall, 1992.

[583] W. Francis and H. Kucera. *Frequency Analysis of English Usage*. Houghton Mifflin Co., 1982.

[584] A. P. Francisco, R. Baeza-Yates, and A. L. Oliveira. Clique analysis of query log graphs. In A. Amir, A. Turpin, and A. Moffat, editors, *SPIRE*, volume 5280 of *Lecture Notes in Computer Science*, pages 188–199, Melbourne, Australia, November 2008. Springer.

[585] K. Franzen and J. Karlgren. Verbosity and interface design. Technical report, Technical Report T2000, 2000.

[586] K. Fredriksson and S. Grabowski. Practical and optimal string matching. In *Proc. SPIRE 2005*, pages 376–387, 2005.

[587] K. Fredriksson and G. Navarro. Average-optimal single and multiple approximate string matching. *ACM Journal of Experimental Algorithmics (JEA)*, 9(1.4), 2004.

[588] H. R. Freitas-Junior, B. A. Ribeiro-Neto, R. de Freitas Vale, A. H. F. Laender, and L. R. S. de Lima. Categorization-driven cross-language retrieval of medical information. *JASIST*, 57(4):501–510, 2006.

[589] L. Freund and E. G. Toms. Enterprise search behaviour of software engineers. In *Proceedings of ACM SIGIR '06*, pages 645–646, New York, NY, USA, 2006. ACM.

[590] L. Freund, E. G. Toms, and C. L. Clarke. Modeling task-genre relationships for IR in the workplace. In *Proceedings of ACM SIGIR '05*, pages 441–448, New York, NY, USA, 2005. ACM.

[591] Y. Freund, R. D. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, pages 170–178, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[592] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, pages 148–156, 1996.

[593] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line

learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997. www.cs.princeton.edu/~schapire/boost.html.

[594] A. Friedlander. D-lib Program: Research in Digital Libraries, May 1998. http://www.dlib.org/.

[595] J. Friedman, R. Kohavi, and Y. Yun. Lazy decision trees. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 717–724. AAAI Press and the MIT Press, Aug. 1996.

[596] Friendster. http://www.friendster.com/, 2002.

[597] N. Fuhr. Models for retrieval with probabilistic indexing. *Information Processing & Management*, 25:55–72, 1989.

[598] N. Fuhr. Optimal polynomial retrieval functions based on the probability ranking principle. *ACM Transactions on Information Systems*, 7(3):183–204, 1989.

[599] N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[600] N. Fuhr. A decision-theoretic approach to database selection in networked ir. *ACM Trans. Inf. Syst.*, 17(3):229–249, 1999.

[601] N. Fuhr and N. Gövert. Retrieval quality vs. effectiveness of specificity-oriented search in XML collections. *Information Retrieval*, 9(1):55–70, 2006.

[602] N. Fuhr, N. Gövert, G. Kazai, and M. Lalmas, editors. *INitiative for the Evaluation of XML Retrieval (INEX). Proceedings of the First INEX Workshop. Dagstuhl, INEX 2002*, ERCIM Workshop Proceedings, Dagstuhl, Germany, 2003. ERCIM.

[603] N. Fuhr and K. Großjohann. XIRQL: An XML query language based on information retrieval concepts. *ACM Transaction on Information Systems*, 22(2):313–356, 2004.

[604] N. Fuhr, P. Hansen, M. Mabe, A. Micsik, and I. Sølvberg. Digital libraries: A generic classification and evaluation scheme. *Lecture Notes in Computer Science*, 2163:187–199, 2001.

[605] N. Fuhr, S. Hartmann, G. Knorz, G. Lustig, M. Schwantner, and K. Tzeras. AIR/X – a rule-based multistage indexing system for large subject fields. In A. Lichnerowicz, editor, *Proceedings of RIAO-91, 3rd International Conference "Recherche d'Information Assistee par Ordinateur"*, pages 606–623, Barcelona, ES, 1991. Elsevier Science Publishers, Amsterdam, NL.

[606] N. Fuhr, J. Kamps, M. . Lalmas, S. Malik, and A. Trotman, editors. *Focused Access to XML Documents, 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007*, Dagstuhl Castle, Germany, 2008. Selected Papers.

[607] N. Fuhr and M. Lalmas. Introduction to the Special Issue on INEX. *Information Retrieval*, 8(4):515–519, 2005.

[608] N. Fuhr and M. Lalmas. Advances in XML retrieval: the INEX initiative. In *Proceedings of the International Workshop on Research Issues in Digital Libraries, IWRIDL 2006*, page 16, Kolkata, India, 2006.

[609] N. Fuhr, M. Lalmas, and S. Malik, editors. *INitiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. Dagstuhl, Germany, December 15-17, 2003*, 2004.

[610] N. Fuhr, M. Lalmas, S. Malik, and G. Kazai, editors. *Advances in XML Information Retrieval and Evaluation: Fourth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2005)*, volume 3977 of *Lecture Notes in Computer Science*. Springer-Verlag, 2006.

[611] N. Fuhr, M. Lalmas, S. Malik, and Z. Szlávik, editors. *Advances in XML Information Retrieval, Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*, volume 3493 of *Lecture Notes in Computer Science*, Dagstuhl Castle, Germany, 2005. Springer. Revised Selected Papers.

[612] N. Fuhr, M. Lalmas, and A. Trotman, editors. *Comparative Evaluation of XML Information Retrieval Systems, 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006*, volume 4518 of *Lecture Notes in Computer Science.* Springer-Verlag, 2007.

[613] N. Fuhr, G. Tsakonas, T. Aalberg, M. Agosti, P. Hansen, S. Kapidakis, C.-P. Klas, L. Kovas, M. Landoni, A. Micsik, C. Papatheodorou, C. Peters, and I. Solvberg. Evaluation of digital libraries. *International Journal of Digital Libraries*, 8(1):21–38, 2007.

[614] G. Furnas, S. Deerwester, S. Dumais, T. Landauer, R. Harshman, L. Streeter, and K. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. In *Proc of the Eleventh Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 465–480, 1988.

[615] D. Gabor. Theory of communication, part iii. *The Journal of the Institute of Electrical Engineers*, pages 429–457, 1946.

[616] V. Gaede and O. Günther. Multidimensional access methods. *ACM Computing Surveys*, 30(2):170–231, 1998.

[617] Q. Gan and T. Suel. Improved techniques for result caching in Web search engines. In J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, editors, *WWW*, pages 431–440, Madrid, Spain, 2009. ACM.

[618] H. Garcia-Molina, L. Gravano, and N. Shivakumar. dSCAM: Finding document copies across multiple databases. In *Proceedings of the Fourth International Conference on Parallel and Distributed Information Systems*, pages 68–79, Miami Beach, FL, USA, December 1996. IEEE Computer Society.

[619] N. Gershon, J. LeVasseur, J. Winstead, J. Croall, A. Pernick, and W. Ruh. Visualizing Internet resources. In *Proceedings '95 Information Visualization*, pages 122–128, Atlanta, Oct. 1995.

[620] S. Geva. GPX - Gardens Point XML IR at INEX 2005. In *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005*, pages 240–253, Dagstuhl Castle, Germany, 2006. Revised Selected Papers.

[621] S. Geva, J. Kamps, and A. Trotman, editors. *Advances in Focused Retrieval, 7th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2008*, volume 5631 of *Lecture Notes in Computer Science*, Dagstuhl Castle, Germany, 2009. Springer. Revised and Selected Papers.

[622] F. C. Gey. Inferring probability of relevance using the method of logistic regression. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Probabilistic Models, pages 222–231, 1994.

[623] S. Ghemawat, H. Gobioff, and S.-T. Leung. The Google file system. In M. L. Scott and L. L. Peterson, editors, *Proceedings of the 19th ACM Symposium on Operating Systems Principles 2003*, pages 29–43, Bolton Landing, NY, USA, October 2003. ACM.

[624] G. Giacinto and F. Roli. Adaptive selection of image classifiers. In *International Conference on Image Analysis and Processing*, pages 38–45, 1997.

[625] D. Gibson, J. Kleinberg, and P. Raghavan. Inferring Web communities from link topologies. In *9th ACM Conference on Hypertext and Hypermedia*, Pittsburgh, USA, 1998.

[626] H. M. Gladney. Trustworthy 100-year digital objects: Evidence after every witness is dead. *ACM Transactions on Information Systems*, 22(3):406–436, 2004.

[627] H. M. Gladney. Principles for digital preservation. *Communications of the ACM*, 49(2):111–116, 2006.

[628] H. M. Gladney and R. A. Lorie. Trustworthy 100-year digital objects: durable encoding for when it's too late to ask. *ACM Transactions on Information Systems*, 23(3):299–324, 2005.

[629] K. Goel, R. Guha, and O. Hansson. Introducing rich snippets. Google Webmaster Central Blog, May 2009. http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html.

[630] S. Goel, A. Broder, E. Gabrilovich, and B. Pang. Anatomy of the long tail: Ordinary people with extraordinary tastes. In *Third ACM Conference on Web Search and Data Mining (WSDM)*, New York, USA, 2010.

[631] N. Goevert, N. Fuhr, M. Lalmas, and G. Kazai. Evaluating the effectiveness of content-oriented XML retrieval methods. *Journal of Information Retrieval*, 9(6):699–722, 2006.

[632] A. Göker and D. He. Analysing Web search logs to determine session boundaries for user-oriented learning. In P. Brusilovsky, O. Stock, and C. Strapparava, editors, *Adaptive Hypermedia*, volume 1892 of *Lecture Notes in Computer Science*, pages 319–322, Trento, Italy, August 2000. Springer.

[633] C. Goldfarb. *The SGML Handbook.* Oxford University Press, Oxford, 1990.

[634] C. Goldfarb and P. Prescod. *The XML Handbook.* Prentice Hall, Oxford, 1998.

[635] S. W. Golomb. Run-length encodings. *IEEE Transactions on Information Theory*, 12(3):399–401, 1966.

[636] G. H. Golub and C. Greif. Arnoldi-type algorithms for computing stationary distribution vectors, with application to pagerank. Technical Report SCCM-04-15, Stanford University, 2004.

[637] B. Gomes. Search quality, continued. *The Official Google Blog*, Jan 2008. http://googleblog.blogspot.com/2008/08/search{}-quality{}-continued.html.

[638] A. Gómez-Pérez, F. Ortíz-Rodríguez, and B. Villazón-Terrazas. Ontology-based legal information retrieval to improve the information access in e-government. In *WWW'06: Proceedings of the 15th international conference on World Wide Web*, pages 1007–1008, New York, NY, USA, 2006. ACM.

[639] M. A. Gonçalves, M. Luo, R. Shen, M. F. Ali, and E. A. Fox. An XML log standard and tool for digital library logging analysis. *Lecture Notes in Computer Science*, 2458:129–143, 2002.

[640] M. A. Gonçalves and E. A. Fox. 5SL – A language for declarative specification and generation of digital libraries. In *Proc. of the 2nd Joint Conf. on Digital Libraries (JCDL'2002)*, pages 263–272, Portland, Oregon, July 14-18, 2002.

[641] M. A. Gonçalves, E. A. Fox, A. Krowne, P. Calado, A. H. F. Laender, A. S. da Silva, and B. A. Ribeiro-Neto. The effectiveness of automatically structured queries in digital libraries. In *JCDL*, pages 98–107, 2004.

[642] M. A. Gonçalves, E. A. Fox, and L. T. Watson. Towards a digital library theory: a formal digital library ontology. *Int. J. on Digital Libraries*, 8(2):91–114, 2008.

[643] M. A. Gonçalves, E. A. Fox, L. T. Watson, and N. A. Kipp. Streams, structures, spaces, scenarios, societies (5s): A formal model for digital libraries. *ACM Transanctions on Information Systems*, 22(2):270–312, 2004.

[644] M. A. Gonçalves, B. Lagoeiro, L. T. Watson, and E. A. Fox. "What is a good

digital library?" - a quality model for digital libraries. *Information Processing & Management*, 43(5), 2007.

[645] M. A. Gonçcalves, G. Panchanathan, U. Ravindranathan, A. Krowne, F. Fox, F. Jagodzinski, and L. Cassel. The XML log standard for digital libraries: analysis, evolution, and deployment. In *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 312–314, Portland, Oregon, 2003.

[646] G. Gonnet. Examples of PAT applied to the Oxford English Dictionary. Technical Report OED-87-02, UW Centre for the New OED and Text Research, Univ. of Waterloo, 1987.

[647] G. Gonnet, R. Baeza-Yates, and T. Snider. New indices for text: Pat trees and Pat arrays. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Data Structures and Algorithms*, pages 66–82. Prentice Hall, Englewood Cliffs, NJ, USA, 1992.

[648] G. Gonnet and F. Tompa. Mind your grammar: a new approach to modeling text. In *Proc. of the Thirteenth Int. Conf. on Very Large Data Bases*, pages 339–346, Brighton, England, Sept 1987.

[649] G. H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures in Pascal and C*. Addison-Wesley, Wokingham, England, 2nd edition, 1991.

[650] Google. Google introduces personalized search services; site enhancements emphasize efficiency. Google Press Center, http://www.google.com/press/pressrel/enhancements.html, March 2004.

[651] Google blog, 2008. http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html.

[652] Google. Advanced search tips. http://www.google.com/support/websearch/bin/answer.py?hl=en\&answer=136861, 2009.

[653] Google notebook. http://www.google.com/notebook, 2009.

[654] Google. Search features. http://www.google.com/help/features.html, 2009.

[655] Google. Search sponsored links. http://www.google.com/sponsoredlinks, 2009.

[656] Google Books. http://books.google.com/, 2009.

[657] Google Scholar. http://scholar.google.com/, 2004.

[658] D. Gorton. Animated Images for a Multimedia Database. Master's thesis, Dept. of Computer Science, Univ. of Maryland, College Park, May 1989.

[659] N. Gövert, M. Abolhassani, N.Fuhr, and K. Großjohann. Content-oriented XML retrieval with HyRex. In *First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, pages 26–32, Dagstuhl, Germany, 2002.

[660] N. Gövert and G. Kazai. Overview of the INitiative for the Evaluation of XML retrieval (INEX) 2002. In *First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, pages 1–17, Dagstuhl, Germany, 2002.

[661] J. Graham. The reader's helper: a personalized document reading environment. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'99)*, pages 481–488, 1999.

[662] M. Granitzer, W. Kienreich, V. Sabol, K. Andrews, and W. Klieber. Evaluating a System for Interactive Exploration of Large, Hierarchically Structured Document Repositories. *Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'04)*, pages 127–133, 2004.

[663] L. Granka, T. Joachims, and G. Gay. Eye-tracking analysis of user behavior in WWW search. *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'04)*, pages 478–479, 2004.

[664] L. Gravano, C.-C. K. Chang, and H. García-Molina. STARTS: Stanford proposal for Internet meta-searching. In *Proc. ACM SIGMOD Inter. Conf. on Management of Data*, pages 207–218, Tucson, AZ, May 1997.

[665] L. Gravano, K. Chang, H. Garcia-Molina, C. Lagoze, and A. Paepcke. STARTS - Stanford protocol proposal for internet retrieval and search. http://infolab.stanford.edu/~gravano/starts.html, January 1997. accessed 15 Jun 2009.

[666] L. Gravano and H. Garcia-Molina. Generalizing GlOSS to vector-space databases and broker hierarchies. In U. Dayal, P. M. D. Gray, and S. Nishio, editors, *VLDB*, pages 78–89, Zurich, Switzerland, September 1995. Morgan Kaufmann.

[667] L. Gravano, H. García-Molina, and A. Tomasic. The effectiveness of GlOSS for the text-database discovery problem. In *Proc. ACM SIGMOD Inter. Conf. on Management of Data*, pages 126–137, Minneapolis, MN, May 1994.

[668] M. Gray. Web characterization studies, 1993.

[669] M. Gray. `www-talk` mailing list, June 1993.

[670] M. Gray. Web growth, 1996.

[671] S. Green. Automated link generation: can we do better than term repetition. In *7th WWW Conference*, Brisbane, Australia, 1998.

[672] S. L. Greene, S. Devlin, P. Cannata, and L. Gómez. No ifs, ands, or ors: A study of database querying. *International Journal of Man [sic] -Machine Studies*, 32(3):303–326, 1990.

[673] G. Grefenstette. Comparing two language identification schemes. In *Proceedings of the 3rd international conference on Statistical Analysis of Textual Data (JADT 1995)*, 1995.

[674] G. Grefenstette. *Cross-Language Information Retrieval*. Kluwer Academic Publishers, Boston, USA, 1998.

[675] G. Grefenstette. Upcoming industrial needs for search. In *Proceedings of ECIR'09*, volume 5478 of *Lecture Notes in Computer Science*, page 3. Springer, 2009.

[676] G. Grefenstette and J. Nioche. Estimation of English and non-English language use on the WWW. In *Proceedings of Content-Based Multimedia Information Access (RIAO)*, pages 237-246, Paris, France, 2000.

[677] G. Griffin, A. D. Holub, and P. Perona. The Caltech-256. Technical report, Caltech, 2006.

[678] D. Griffiths. A pragmatic approach to Spearman's rank correlation coefficient. *Teaching Statistics*, 2:10–13, 1980.

[679] G. Grinstein, T. O'Connell, S. Laskowski, C. Plaisant, J. Scholtz, and M. Whiting. VAST 2006 Contest–A Tale of Alderwood. *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST'06)*, pages 215–216, 2006.

[680] R. Grossi, A. Gupta, and J. S. Vitter. High-order entropy-compressed text indexes. In *SODA*, pages 841–850, 2003.

[681] R. Grossi and J. Vitter. Compressed suffix arrays and suffix trees with applications to text indexing and string matching. *SIAM Journal on Computing*, 35(2):378–407, 2006. Preliminary version in Proc. 32nd ACM Symposium on Theory of Computing (STOC), pp. 397–406.

[682] D. A. Grossman and O. Frieder. *Information Retrieval: Algorithms and Heuristics*. Kluwer Academic Publishers, 1998.

[683] V. Guduvada, V. Raghavan, W. Grosky, and R. Kasanagottu. Information retrieval on the world wide web. *IEEE Internet Computing*, Oct-Nov:58–68, 1997.

[684] J. Guiver and E. Snelson. Learning to rank with softrank and gaussian processes. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 259–266, New York, NY, USA, 2008. ACM Press.

[685] A. Gulli and A. Signorini. The indexable Web is more than 11.5 billion pages. In *Poster proceedings of the 14th international conference on World Wide Web*, pages 902–903, Chiba, Japan, 2005. ACM Press.

[686] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram. XRANK: Ranked Keyword Search over XML Documents. In *SIGMOD Conference*, pages 16–27, 2003.

[687] V. Gupta and R. H. Campbell. Internet search engine freshness by Web server help. In *Proceedings of the Symposium on Internet Applications (SAINT)*, pages 113–119, San Diego, California, USA, 2001.

[688] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proc. ACM SIGMOD*, pages 47–57, Boston, Mass, June 1984.

[689] J. Gwertzman and M. Seltzer. World-wide Web cache consistency. In *Proceedings of the 1996 Usenix Technical Conference*, San Diego, California, USA, January 1996.

[690] Z. Gyöngyi and H. Garcia-Molina. Web spam taxonomy. In *First International Workshop on Adversarial Information Retrieval on the Web*, 2005.

[691] Hadoop. http://hadoop.apache.org/, 2007.

[692] C. D. Hafner. Representation of knowledge in a legal information retrieval system. In *SIGIR '80: Proceedings of the 3rd annual ACM conference on Research and development in information retrieval*, pages 139–153, Kent, UK, UK, 1981. Butterworth & Co.

[693] D. Haines and W. B. Croft. Relevance feedback and inference networks. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Inference Networks, pages 2–11, 1993.

[694] H. Han, C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. A. Fox. Automatic document metadata extraction using support vector machines. In *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2003*, pages 37–48. IEEE Computer Society, 2003.

[695] H. Han, L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis. Two supervised learning approaches for name disambiguation in author citations. In *JCDL'04: Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 296–305, Tucson, Arizona, USA, 2004.

[696] U. Hanani, B. Shapira, and P. Shoval. Information filtering: Overview of issues, research and systems. *User Modeling and User-Adapted Interaction (UMUAI)*, 11(3):203–259, 2001.

[697] P. Hansen and K. Järvelin. The information seeking and retrieval process at the swedish patent- and registration office. In *Proceedings of the ACM SIGIR'2000 workshop on Patent Retrieval*, July 2000. http://www.sics.se/humle/projects/pir/patent.html.

[698] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybertinetics*, 3(6):610–621, 1973.

[699] E. Hargittai. Classifying and Coding Online Actions. *Social Science Computer Review*, 22(2):210–227, 2004.

[700] D. Harman. Ranking algorithms. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Data Structures & Algorithms*, pages 363–392. Prentice Hall, 1992.

[701] D. Harman. Relevance feedback and other query modification techniques. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Data Structures & Algorithms*, pages 241–263. Prentice Hall, 1992.

[702] D. Harman. Overview of the first text retrieval conference (TREC-1). In D. Harman, editor, *Proceedings of the First Text REtrieval Conference (TREC-1)*, pages 1–20. NIST Special Publication 500-207, 1993.

[703] D. Harman. Overview of the second text retrieval conference (TREC-2). In D. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*. NIST Special Publication, 1994.

[704] D. Harman. Overview of the third text retrieval conference (TREC-3). In D. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication, 1995.

[705] D. Harman, E. Fox, R. Baeza-Yates, and W. Lee. Inverted files. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Algorithms and Data Structures*, chapter 3, pages 28–43. Prentice-Hall, Englewood Cliffs, NJ, USA, 1992.

[706] S. Harnad. The self-archiving initiative. *Nature*, 410, 2001.

[707] D. Harper. *Relevance Feedback in Document Retrieval Systems: An Evaluation of Probabilistic Strategies.* PhD thesis, Jesus College, Cambridge, England, 1980.

[708] D. Harper and C. van Rijsbergen. An evaluation of feedback in document retrieval using co-occurrence data. *Journal of Documentation*, 34(3):189–216, 1978.

[709] S. P. Harter and C. A. Hert. Evaluation of information retrieval systems: Approaches, issues, and methods. *Review of Information Science and Technology (ARIST)*, 32:3–94, 1997.

[710] S. Harum. Digital Library Initiative, January 1998. http://dli.grainger.uiuc.edu/national.htm.

[711] B. G. Haskell, A. Puri, and A. N. Netravali. *Digital Video: An introduction to MPEG-2.* Kindle Book, 12 1996.

[712] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Machine Learning.* Springer, 2008. 2nd Edition.

[713] A. Hatter and E. Trapasso. Managers say the majority of information obtained for their work is useless, accenture survey finds. accenture.tekgroup.com/article_display.cfm?article_id=4484, January 2007.

[714] C. Hauff and L. Azzopardi. When is query performance prediction effective? In J. Allan, J. A. Aslam, M. Sanderson, C. Zhai, and J. Zobel, editors, *SIGIR*, pages 829–830, Boston, MA, USA, 2009. ACM.

[715] C. Hauff, L. Azzopardi, and D. Hiemstra. The combination and evaluation of query performance prediction methods. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, 2009.

[716] C. Hauff, D. Hiemstra, and F. de Jong. A survey of pre-retrieval query performance predictors. In J. G. Shanahan, S. Amer-Yahia, I. Manolescu, Y. Zhang, D. A. Evans, A. Kolcz, K.-S. Choi, and A. Chowdhury, editors, *CIKM*, pages 1419–1420, Napa Valley, California, USA, 2008. ACM.

[717] C. Hauff, V. Murdock, and R. Baeza-Yates. Improved query difficulty prediction for the web. In J. G. Shanahan, S. Amer-Yahia, I. Manolescu, Y. Zhang, D. A. Evans, A. Kolcz, K.-S. Choi, and A. Chowdhury, editors, *CIKM*, pages 439–448, Napa Valley, CA, USA, 2008. ACM.

[718] T. Haveliwala. Efficient computation of pagerank. Technical report, Stanford

University, 1999.

[719] D. Hawking. Challenges in enterprise search. In *Proceedings of the Australasian Databases Conference ADC2004*, pages 15–26, Dunedin, New Zealand, January 2004. Australian Computer Society. Invited paper: http://es.csiro.au/pubs/hawking_adc04keynote.pdf.

[720] D. Hawking, N. Craswell, F. Crimmins, and T. Upstill. How valuable is external link evidence when searching enterprise webs? In *Proceedings of the Australasian Database Conference ADC2004*, pages 77–84, January 2004. http://es.csiro.au/pubs/hawking_adc04.pdf.

[721] D. Hawking, T. Rowlands, and M. Adcock. Improving rankings in small-scale Web search using click-implied descriptions. *Australian Journal of Intelligent Information Processing Systems. ADCS 2006 special issue.*, 9(2):17–24, December 2006. http://es.csiro.au/pubs/hawking-rowlands-adcock-adcs2006.pdf.

[722] D. Hawking, T. Rowlands, and P. Thomas. C-test: Supporting novelty and diversity in testfiles for search evaluation. In *Proceedings of the SIGIR workshop on redundancy, diversity and interdependent document relevance*, 2009. http://david-hawking.net/pubs/hawking-rowlands-thomas09.pdf.

[723] D. Hawking, E. Voorhees, N.Craswell, and P. Bailey. Overview of the TREC-8 Web Track. In *The Eighth Text REtrieval Conference (TREC-8), Gaithersburg, Maryland, National Institute of Standards and Technology (NIST)*, 1999.

[724] D. Hawking and J. Zobel. Does topic metadata help with Web search? *JASIST*, 58(5):613–628, 2007. Preprint at http://es.csiro.au/pubs/hawking_zobel_jasist.pdf.

[725] Hbase. http://hadoop.apache.org/hbase/, 2008.

[726] HDFS Architecture. http://hadoop.apache.org/common/docs/current/hdfs_design.html, 2008.

[727] B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. In *The Eleventh Symposium on String Processing and Information Retrieval (SPIRE)*, pages 43–54, 2004.

[728] D. He, A. Göker, and D. J. Harper. Combining evidence for automatic Web session identification. *Information Processing & Management*, 38(5):727–742, 2002.

[729] J. He, M. Larson, and M. de Rijke. Using coherence-based measures to predict query difficulty. In *Advances in Information Retrieval: 29th European Conference on IR Research*, pages 689–694, 2008.

[730] H. Heaps. *Information Retrieval - Computational and Theoretical Aspects*. Academic Press, 1978.

[731] M. Hearst. Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd Meeting of the Association for Computational Linguistics (ACL'94)*, pages 9–16, Las Cruces, NM, June 1994.

[732] M. Hearst. TileBars: Visualization of Term Distribution Information in Full Text Information Access. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'95)*, Denver, CO, May 1995.

[733] M. Hearst. Improving full-text precision using simple query constraints. In *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval (SDAIR'96)*, Las Vegas, NV, 1996.

[734] M. Hearst. TextTiling: Segmenting Text into Multi-Paragraph Subtopic Passages. *Computational Linguistics*, 23(1):33–64, 1997.

[735] M. Hearst. *Search User Interfaces*. Cambridge University Press, 2009.

[736] M. Hearst, A. Divoli, H. Guturu, A. Ksikes, P. Nakov, M. Wooldridge, and J. Ye. BioText Search Engine: beyond abstract search. *Bioinformatics*, 23(16):2196, 2007.

[737] M. Hearst, J. English, R. Sinha, K. Swearingen, and K.-P. Yee. Finding the flow in Web site search. *Communications of the ACM*, 45(9), September 2002.

[738] M. A. Hearst. Clustering versus faceted categories for information exploration. *Commun. ACM*, 49(4):59–61, 2006.

[739] M. A. Hearst. Design recommendations for hierarchical faceted search interfaces. In A. Z. Broder and Y. S. Maarek, editors, *Proceedings of the SIGIR 2006 Workshop on Faceted Search*, pages 26–30, August 2006.

[740] J. Heer and E. H. Chi. Separating the swarm: categorization methods for user sessions on the web. In *CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 243–250, New York, NY, USA, 2002. ACM.

[741] T. Heimonen and N. Jhaveri. Visualizing Query Occurrence in Search Result Lists. *Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'05)*, pages 877–882, 2005.

[742] N. Heintze. Scalable document fingerprinting. In *1996 USENIX Workshop on Electronic Commerce*, November 1996.

[743] J. M. Hellerstein, J. F. Naughton, and A. Pfeffer. Generalized search trees for database systems. In *Proc. of VLDB Conf.*, pages 562–573, Zurich, Switzerland, Sept 1995.

[744] M. Hemmje, C. Kunkel, and A. Willett. LyberWorld – a visualization user interface supporting fulltext retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'94)*, pages 249–259, Dublin, Ireland, July 1994.

[745] M. Henzinger. Web information retrieval - an algorithmic perspective. In *European Symposium on Algorithms*, pages 1–8, 2000. http://citeseer.nj.nec.com/571448.html.

[746] M. Henzinger. Hyperlink analysis for the web. *IEEE Internet Computing*, 5(1):45–50, 2001.

[747] M. R. Henzinger, R. Motwani, and C. Silverstein. Challenges in Web search engines. *SIGIR Forum*, 36(2):11–22, 2002.

[748] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In Smola, Bartlett, Schoelkopf, and Schuurmans, editors, *Advances in Large Margin Classifiers*. MIT Press, Cambridge, MA, 2000.

[749] W. Hersh. *Information Retrieval – A Health and Biomedical Perspective*. Springer, 2009. Third Edition.

[750] W. Hersh, C. Buckley, T. J. Leone, and D. Hickam. Ohsumed: an interactive retrieval evaluation and new large test collection for research. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–201. Springer-Verlag New York, Inc., 1994.

[751] W. R. Hersh. Improving health care through information. *Jama*, 288:1955–1958, 2002.

[752] W. R. Hersh. Health care information technology—progress and barriers. *Jama*, 292:2273–2274, 2004.

[753] W. R. Hersh and D. H. Hickam. How well do physicians use electronic information retrieval systems? a framework for investigation and systematic review. *Jama*, 280:1347–1352, 1998.

[754] M. Hersovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur.

The shark-search algorithm. An application: tailored Web site mapping. In *Proceedings of the seventh conference on World Wide Web*, pages 317–326, Brisbane, Australia, April 1998. Elsevier Science.

[755] M. Hertzum and E. Frokjaer. Browsing and querying in online documentation: A study of user interfaces and the interaction process. *ACM Transactions on Computer-Human Interaction (ToCHI)*, 3(2):136–161, 1996.

[756] M. Hertzum and A. M. Pejtersen. The information-seeking practices of engineers: searching for documents as well as for people. *Information Processing and Management*, 36:761–778, 2000.

[757] E. v. Herwijnen. *Practical SGML*. Kluwer Academic Publishers, second edition edition, 1994.

[758] E. Hetzler and A. Turner. Analysis Experiences Using Information Visualization. *IEEE Computer Graphics and Applications*, 24(5):22–26, 2004.

[759] A. Heydon and M. Najork. Mercator: A scalable, extensible Web crawler. *World Wide Web Conference*, 2(4):219–229, April 1999.

[760] D. Hiemstra. A linguistically motivated probabilistic model of information retrieval. In *Proceedings of the Second European Conference on Research and Advance Technology for Digital Libraries (ECDL)*, pages 569–584, 1998.

[761] D. Hiemstra and R. Baeza-Yates. Structured text retrieval models. In *Encyclopedia of Database Systems*. Springer, 2009.

[762] D. Hiemstra and W. Kraaij. Twenty-one at TREC-7: Ad-hoc and cross-language track. In *Proceedings of the Seventh Text Retrieval Conference (TREC-7)*, pages 227–238, 1999.

[763] C. Hildreth. OPAC research: laying the groundwork for future OPAC design. *The Online Catalogue: Development and Directions*, pages 1–24, 1989.

[764] C. Hildreth. Online catalog design models: Are we moving in the right direction? Report Submitted to the Council on Library and Information Resources, 1995, updated 2000. http://myweb.cwpost.liu.edu/childret/clr-opac.html.

[765] L. L. Hill, G. Jance, R. Dolin, J. Frew, and M. Larsgaard. Collection metadata solutions for digital library applications. *JASIS*, 50(13):1169–1181, 1999.

[766] W. Hill, J. Hollan, D. Wroblewski, and T. McCandless. Edit wear and read wear. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'92)*, 92:3–9, 1992.

[767] R. Himmeroder, G. Lausen, B. Ludascher, and C. Schlepphorst. On a declarative semantics for Web queries. In *Int. Conf. on Deductive and Object-Oriented Database (DOOD)*, pages 386–398, Singapore, December 1997.

[768] J. Hirai, S. Raghavan, H. Garcia-Molina, and A. Paepcke. Webbase: a repository of Web pages. *Computer Networks (Amsterdam, Netherlands: 1999)*, 33(1–6):277–293, 2000.

[769] D. S. Hirschberg and D. A. Lelewer. Efficient decoding of prefix codes. *Communications of the ACM*, 33(4), 1990.

[770] O. Hoeber and X. D. Yang. A comparative user study of Web search interfaces: HotMap, Concept Highlighter, and Google. *IEEE/WIC/ACM International Conference on Web Intelligence*, 2006.

[771] C. Hölscher and G. Strube. Web search behavior of Internet experts and newbies. *Computer Networks*, 33(1-6):337–346, 2000.

[772] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, Reading, Mass., 1979.

[773] K. Hornbæk and E. Frøkjær. Do Thematic Maps Improve Information Re-

trieval. *Human-Computer Interaction (INTERACT'99)*, pages 179–186, 1999.

[774] R. N. Horspool. Practical fast searching in strings. *Software Practice and Experience*, 10:501–506, 1980.

[775] R. N. Horspool and G. V. Cormack. Constructing word-based text compression algorithms. In *Proc. of IEEE Second Data Compression Conference*, pages 62–81, 1992.

[776] E. Hörster, R. Lienhart, and M. Slaney. Image retrieval on large-scale image databases. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval CIVR 07*, July 2007.

[777] M. Hosseini. A study on performance volatility in information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (doctoral consortium)*, page 853, Boston, MA, USA, 2009.

[778] A. J. M. Houtsma. Pitch and timbre: Definition, meaning and use. *Journal of New Music Research*, 26:104–115, 1997.

[779] P. Howarth and S. Rüger. Robust texture features for still-image retrieval. *IEE Proceedings on Vision, Image and Signal*, 152(6):868–874, 2005.

[780] D. Howe and H. Nissenbaum. Trackmenot: Resisting surveillance in Web search. In I. Kerr, C. Lucock, and V. Steeves, editors, *On the Identity Trail: Privacy, Anonymity and Identity in a Networked Society*. Oxford: Oxford University Press, 2009.

[781] C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425, Mar. 2002.

[782] W. Hsu, L. Kennedy, and S.-F. Chang. Reranking methods for visual search. *Multimedia, IEEE*, 14(3):14–22, July-Sept. 2007.

[783] ht://Dig. http://www.htdig.org/, 2007.

[784] Y. Hu, H. Li, Y. Cao, D. Meyerzon, and Q. Zheng. Automatic extraction of titles from general documents using machine learning. In *JCDL'05: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, Tools & techniques: supporting classification, pages 145–154, 2005.

[785] F. Huang, S. Watt, D. Harper, and M. Clark. Compact representations in XML retrieval. In *Comparative Evaluation of XML Information Retrieval Systems, 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006*, pages 64–72, Dagstuhl Castle, Germany, 2006. Revised and Selected Papers.

[786] J. Huang, S. Ertekin, Y. Song, H. Zha, and C. L. Giles. Efficient multiclass boosting classification with active learning. In *SDM*. SIAM, 2007. http://www.siam.org/meetings/proceedings/2007/datamining/papers/027Huang.pdf.

[787] J. Huang, S. Kumar, M. Mitra, W. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 762–768, 1997.

[788] X. Huang, A. Acero, and H.-W. Hon. *Spoken language processing*. Prentice Hall PTR, 2000.

[789] X. Huang, F. Peng, A. An, and D. Schuurmans. Dynamic Web log session identification with statistical language models. *JASIST*, 55(14):1290–1303, 2004.

[790] Z. Huang, W. Chung, T.-H. Ong, and H. Chen. A graph-based recommender system for digital library. In *Proceedings of the Second ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 65–73, Portland, Oregon, 2002.

[791] Z. Huang, X. Li, and H. Chen. Link prediction approach to collaborative filtering. In *Proceedings ACM/IEEE Joint Conference on Digital Libraries, JCDL 2005*, pages 141–142, Denver, CA, USA, 2005.

[792] B. A. Huberman. *The Laws of the Web: Patterns in the Ecology of Information.* The MIT Press, October 2001.

[793] B. A. Huberman and L. A. Adamic. Evolutionary dynamics of the World Wide Web. *Condensed Matter*, January 1999.

[794] B. A. Huberman and L. A. Adamic. Growth dynamics of the world-wide web. *Nature*, 399, 1999.

[795] D. Huffman. A method for the construction of minimum-redundancy codes. *Proc. of the I.R.E.*, 40(9):1090–1101, 1952.

[796] D. Hull. Using statistical testing in the evaluation of retrieval experiments. In *SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 329–338, 1993.

[797] D. A. Hull. Improving text retrieval for the routing problem using latent semantic indexing. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 282–291, Dublin, Ireland, July 1994.

[798] J. Hunter and L. Armstrong. A comparison of schemas for video metadata representation. In *WWW'99: Proceedings of the eighth international conference on World Wide Web*, pages 1431–1451, New York, NY, USA, 1999. Elsevier North-Holland, Inc.

[799] T. Hybrid Library. The HyLIFE Hybrid Library Toolkit, 2001. http://hylife.unn.ac.uk/toolkit/.

[800] Hypertable. http://www.hypertable.org/, 2009.

[801] IAB. Interactive advertising bureau, September 2006. http://www.iab.net/news/pr_2006_05_30.asp.

[802] R. Iannella. Digital rights management (DRM) architectures. *D-Lib Magazine*, 7, 2001.

[803] Ibiblio. Internet Pioneers: Tim Berners-Lee. Ibiblio, the Public's Library and Digital Archive, http://www.ibiblio.org/pioneers/lee.html, September, 2006.

[804] IBM OmniFind Yahoo! Edition. http://omnifind.ibm.yahoo.net/, 2007.

[805] IDC. The Enterprise Workplace: How It Will Change the Way We Work. IDC Report 32919, February 2005.

[806] E. Ide. New experiments in relevance feedback. In G. Salton, editor, *The SMART Retrieval System*, pages 337–354. Prentice Hall, 1971.

[807] IEEE Standards Committee on Optical Disk and Multimedia Platforms (SCODMP). IEEE SFQL. Technical report, IEEE, Washington, USA, 1992.

[808] IEEE Computer, February 1999. 32(2).

[809] Indri. http://www.lemurproject.org/indri/, 2007.

[810] P. Ingwersen and K. Jarvelin. *The Turn: Integration of Information Seeking and Retrieval in Context.* Springer, 2005.

[811] Inktomi. http://www.inktomi.com, 1998.

[812] Internet Systems Consortium. Internet domain survey. http://ftp.isc.org/www/survey/reports/current/, 2009.

[813] Information Processing & Management, March 1999. 35(3).

[814] K. Itakura and C. A. Clarke. University of Waterloo at INEX2007: Adhoc and Link-the-Wiki Tracks. In *Focused Access to XML Documents, 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007*, pages 417–425, Dagstuhl Castle, Germany, 2007. Selected Papers.

[815] IWS. Internet world stats, January 2010. http://www.internetworldstats.com/top20.htm.

[816] P. Jaccard. étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la Socit Vaudoise des Sciences Naturelles*, 37:547579, 1901.

[817] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.

[818] S. C. Jane Hunter. Implementing Preservation Strategies for Complex Multimedia Objects. In *Proc. 7th European Conf. Research and Advanced Technology for Digital Libraries, ECDL 2003*, pages 473–486, Trodheim, Norway, August 17-22, 2003.

[819] B. Jansen, A. Spink, and S. Koshman. Web searcher interaction with the Dogpile.com metasearch engine. *Journal of the American Society for Information Science and Technology*, 58(5):744–755, 2007.

[820] B. Jansen, A. Spink, and J. Pedersen. A Temporal Comparison of AltaVista Web Searching. *Journal of the American Society for Information Science and Technology*, 56(6):559–570, 2005.

[821] B. J. Jansen. *Understanding User-Web Interactions via Web Analytics.* Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan Claypool, 2009.

[822] B. J. Jansen, D. L. Booth, and A. Spink. Determining the user intent of Web search engine queries. In *Proc. of the 16th international conference on World Wide Web*, pages 1149–1150. ACM Press, 2007.

[823] B. J. Jansen and A. Spink. An analysis of Web searching by European AlltheWeb.com users. *Information Processing and Management: an International Journal*, 41(2):361–381, 2005.

[824] B. J. Jansen and A. Spink. How are we searching the World Wide Web? a comparison of nine search engine transaction logs. *Information Processing & Management*, 42(1):248-263, 2006.

[825] B. J. Jansen, A. Spink, C. Blakely, and S. Koshman. Defining a session on Web search engines. *JASIST*, 58(6):862-871, 2007.

[826] B. P. Jansen, A. Spink, J. Bateman, and T. Saracevic. Real life information retrieval: A study of user queries on the web. *ACM SIGIR Forum*, 32(1):5–17, Spring 1998.

[827] N. Jardine and C. Rijsbergen. The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval*, 7:217–240, 1971.

[828] K. Järvelin and J. Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In *ACM SIGIR International Conference on Information Retrieval*, pages 41–48, 2000.

[829] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[830] N. Jayant, J. Johnston, and R. Safranek. Signal compression based on models of human perception. *Proceedings of the IEEE*, 81(10), October 1993.

[831] G. Jeh and J. Widom. Scaling personalized Web search. In *WWW'03: Proceedings of the 12th international conference on World Wide Web*, pages 271–279, New York, NY, USA, 2003. ACM.

[832] C. Jenkins, C. Corritore, and S. Wiedenbeck. Patterns of information seeking

on the Web: a qualitative study of domain expertise and Web expertise. *IT & Society*, 1(3):64–89, 2003.

[833] B.-S. Jeong and E. Omiecinski. Inverted file partitioning schemes in multiple disk systems. *IEEE Trans. Par. and Dist. Syst.*, 6(2):142-153, Feb. 1995.

[834] S. Ji, G. Li, C. Li, and J. Feng. Efficient interactive fuzzy keyword search. In *WWW'09: Proceedings of the 18th international conference on World wide web*, pages 371–380, New York, NY, USA, 2009. ACM.

[835] Y. Jing and S. Baluja. PageRank for product image search. In *WWW'08: Proceeding of the 17th International Conference on World Wide Web*, pages 307–316, New York, NY, USA, 2008. ACM.

[836] J.Nielsen. *Hypertext and Hypermedia.* Academic Press, 1990.

[837] T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *International Conference on Machine Learning (ICML)*, pages 143–151, 1997.

[838] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA, 1999.

[839] T. Joachims. SVMLight – Support Vector Machine, 1999. http://svmlight.joachims.org/.

[840] T. Joachims. SVMPerf - Support Vector Machine, 1999. http://svmlight.joachims.org/svm_perf.html.

[841] T. Joachims. Optimizing search engines using clickthrough data. In D. Hand, D. Keim, and R. Ng, editors, *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-02)*, pages 132–142, Edmonton, Alberta, Canada, July 2002.

[842] T. Joachims. Evaluating retrieval performance using clickthrough data. In J. Franke, G. Nakhaeizadeh, and I. Renz, editors, *Text Mining*, pages 79–96. Physica/Springer Verlag, 2003.

[843] T. Joachims. Training linear SVMs in linear time. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 217–226, 2006.

[844] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 154–161, New York, NY, USA, 2005. ACM.

[845] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in Web search. *ACM Transactions on Information Systems*, 25(2), 2007.

[846] T. Joachims and F. Radlinski. Search engines that learn from implicit feedback. *IEEE Computer*, 40(8):34–40, August 2007.

[847] R. Jones and K. L. Klinkner. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In J. G. Shanahan, S. Amer-Yahia, I. Manolescu, Y. Zhang, D. A. Evans, A. Kolcz, K.-S. Choi, and A. Chowdhury, editors, *CIKM*, pages 699–708, Napa Valley, CA, USA, November 2008. ACM.

[848] R. Jones, R. Kumar, B. Pang, and A. Tomkins. "I know what you did last summer": query logs and user privacy. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 909–914, New York, NY, USA, 2007. ACM Press.

[849] R. Jones, R. Kumar, B. Pang, and A. Tomkins. Vanity fair: privacy in query log bundles. In *CIKM '08: Proceeding of the 17th ACM conference on Information*

and knowledge management, pages 853–862, New York, NY, USA, 2008. ACM Press.

[850] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *WWW'06: Proceedings of the 15th international conference on World Wide Web*, pages 387–396, New York, NY, USA, 2006. ACM Press.

[851] S. Jones. Graphical query specification and dynamic result previews for a digital library. In *Proceedings of the 11th annual ACM symposium on User Interface Software and Technology (UIST'98)*, pages 143–151, San Francisco, USA, November 1998.

[852] W. Jones, H. Bruce, and S. Dumais. Keeping Found Things Found on the Web. *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM'01)*, pages 119–126, 2001.

[853] W. Jones, S. Dumais, and H. Bruce. Once Found, What Then? A Study of "Keeping" Behaviors in the Personal Use of Web Information. *Proceedings of the American Society for Information Science and Technology*, 39(1):391–402, 2002.

[854] D. Jonker, W. Wright, D. Schroh, P. Proulx, and B. Cort. Information Triage with TRIST. *Proceedings of the International Conference on Intelligence Analysis*, 2005.

[855] W.-K. Joo and S. H. Myaeng. Improving retrieval effectiveness with hyperlink information. In *Proceedings of International Workshop on Information Retrieval with Asian Languages (IRAL)*, Singapore, October 1998.

[856] T. Joyce and R. Needham. The thesaurus approach to information retrieval. In K. S. Jones and P. Willet, editors, *Readings in Information Retrieval*, pages 15–20. Morgan Kaufmann Publishers, Inc., 1997.

[857] F. Junqueira and K. Marzullo. Coterie availability in sites. In *Proceedings of the International Conference on Distributed Computing (DISC)*, number 3724 in LNCS, pages 3–17, Krakow, Poland, September 2005. Springer Verlag.

[858] S. Kaasten, S. Greenberg, and C. Edwards. How People Recognise Previously Seen Web Pages from Titles, URLs and Thumbnails. *People and Computers*, pages 247–266, 2002.

[859] B. Kahle. Archiving the Internet. http://www.alexa.com/~brewster/essays/sciam_article.html, 1997.

[860] B. Kahle and A. Medlar. An information server for corporate users: Wide Area Information Servers. *ConneXions - the Interoperability Report*, 5(11):2–9, 1991. ftp://think.com/wais/wais-corporate-paper.text.

[861] M. Kaisser, M. Hearst, and J. Lowe. Improving Search Results Quality by Customizing Summary Lengths. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technlogies (ACL-HLT'08)*, 2008.

[862] M. Käki. Findex: Search Result Categories Help Users When Document Ranking Fails. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'05)*, pages 131–140, 2005.

[863] J. Kalbach. *Designing Web navigation*. O'Reilly, 2007.

[864] T. Kalt. A new probabilistic model of text classification and retrieval. Technical Report IR-78, CIIR, Univ. of Massachussets at Amherst, 1996. http://ciir.cs.umass.edu/~kalt/kaltTr96.pdf.

[865] I. Kamel and C. Faloutsos. Hilbert R-tree: An improved R-tree using fractals. In *Proc. of VLDB Conference*, pages 500–509, Santiago, Chile, Sept 1994.

[866] J. Kamps, M. de Rijke, and B. Sigurbjörnsson. Length normalization in XML retrieval. In *27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK*, pages 80–87, 2004.

[867] J. Kamps, J. Pehcevski, G. Kazai, M. Lalmas, and S. Robertson. INEX 2007 Evaluation Metrics. In *Focused access to XML documents, 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007*, Dagstuhl Castle, Germany, 2008. Selected Papers.

[868] J. Kamps and B. Sigurbjörnsson. What do users think of an XML element retrieval system? In *Advances in XML Information Retrieval and Evaluation: Fourth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2005)*, volume 3977 of *Lecture Notes in Computer Science*, pages 411–421, 2006.

[869] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub. Exploiting the block structure of the Web for computing pagerank, 2003.

[870] S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub. Extrapolation methods for accelerating pagerank computations. In *Proceedings of the twelfth international conference on World Wide Web*, pages 261–270. ACM Press, 2003.

[871] B. Kang and R. Wilensky. Toward a model of self-administering data. In *Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 322–330, Roanoke, Virginia, 2001.

[872] I.-H. Kang and G. Kim. Query type classification for Web document retrieval. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 64–71, New York, NY, USA, 2003. ACM Press.

[873] T. Kanungo and D. Orr. Predicting the readability of short Web summaries. In *ACM WSDM '09: 2nd ACM International Conference on Web Search and Data Mining*, 2009.

[874] J. Kapms. Indexing units. In *Encyclopedia of Database Systems*. Springer, 2009.

[875] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin. Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web. In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 654–663, El Paso, TX, USA, 1997. ACM.

[876] J. Kärkkäinen and P. Sanders. Simple linear work suffix array construction. In *Proc. ICALP'03*, pages 943–955, 2003.

[877] R. Karp and M. Rabin. Efficient randomized pattern-matching algorithms. *IBM Journal of Research and Development*, 31(2):249–260, Mar. 1987.

[878] M. Kaszkiel and J. Zobel. Passage retrieval revisited. In *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185. ACM Press, 1997.

[879] N. Katayama and S. Satoh. The SR-tree: An index structure for high-dimensional nearest neighbor queries. In *Proc. of ACM SIGMOD*, pages 369–380, Tucson, AZ, 1997.

[880] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, March 1953.

[881] H. Kautz, B. Selman, and M. Shah. The hidden Web. *AI Magazine*, 18(2):27–36, 1997.

[882] G. Kazai. Choosing an Ideal Recall-Base for the Evaluation of the Focused Task: Sensitivity Analysis of the XCG Evaluation Measures. In *Comparative Evaluation of XML Information Retrieval Systems, 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006*, pages 35–44, Dagstuhl Castle, Germany, 2007. Revised and Selected Papers.

[883] G. Kazai. INitiative for the Evaluation of XML retrieval (INEX). In *Encyclopedia of Database Systems*. Springer, 2009.

[884] G. Kazai and A. Doucet. Overview of the INEX 2007 Book Search Track (BookSearch '07). *SIGIR Forum*, 42(1):2–15, 2008.

[885] G. Kazai and M. Lalmas. eXtended Cumulated Gain Measures for the Evaluation of Content-oriented XML Retrieval. *ACM Transactions on Information Systems*, 24(4):503–542, 2006.

[886] G. Kazai, M. Lalmas, and A. de Vries. The overlap problem in content-oriented XML retrieval evaluation. In *27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK*, pages 72–79, 2004.

[887] G. Kazai, M. Lalmas, and J. Reid. Construction of a test collection for the focused retrieval of structured documents. In *Advances in Information Retrieval, 25th European Conference on IR Research, ECIR 2003, Pisa, Italy*, pages 88–103, 2003.

[888] G. Kazai, N. Milic-Frayling, and J. Costello. Towards methods for the collective gathering and quality control of relevance assessments. In *ACM SIGIR International Conference on Information Retrieval*, 2009.

[889] G. Kazai and T. Rölleke. A Scalable Architecture for XML Retrieval. In *First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, pages 49–56, Dagstuhl, Germany, 2002.

[890] Y. Ke, L. Deng, W. Ng, and D.-L. Lee. Web dynamics and their ramifications for the development of Web search engines. *Computer Networks*, 50(10):1430–1447, July 2006.

[891] KEA: Keyphrase extraction algorithm. http://www.nzdl.org/Kea/, 2009.

[892] J. Kekäläinen. Binary and graded relevance in IR evaluations: comparison of the effects on ranking of IR systems. *Information Processing & Management*, 41(5):1019–1033, 2005.

[893] J. Kekäläinen, P. Arvola, and M. Junkkari. Contextualization. In *Encyclopedia of Database Systems*. Springer, 2009.

[894] D. Kelly. Methods for evaluating interactive information retrieval systems with users. *Foundations and Trends in Information Retrieval*, 3(1-2):1–224, 2009.

[895] D. Kelly, V. Dollu, and X. Fu. The Loquacious User: A Document-Independent Source of Terms for Query Expansion. *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'05)*, pages 457–464, 2005.

[896] D. Kelly and J. Teevan. Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum*, 37(2):18–28, 2003.

[897] K. Kelly. Scan This Book!, May 2006. New York Times Magazine.

[898] M. G. Kendall. *Rank Correlation Methods*. Hafner Publishing Company, 1955.

[899] R. Kengeri, C. D. Seals, H. D. Harley, H. P. Reddy, and E. A. Fox. Usability study of digital libraries: ACM, IEEE-CS, NCSTRL, NDLTD. *Int. J. on Digital Libraries*, 2(2-3):157–169, 1999.

[900] L. Kennedy, M. Naaman, S. Ahern, R. Nair, and T. Rattenbury. How Flickr helps us make sense of the world: Context and content in community-contributed media collections. In *MULTIMEDIA '07: Proceedings of the 15th International Conference on Multimedia*, pages 631–640, New York, NY, USA, 2007. ACM.

[901] L. S. Kennedy, S.-F. Chang, and I. V. Kozintsev. To search or to label?: Predicting the performance of search-based automatic image classifiers. In *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia IR*, pages 249–258, New York, NY, USA, 2006. ACM.

[902] L. S. Kennedy and M. Naaman. Generating diverse and representative image search results for landmarks. In *WWW'08: Proceeding of the 17th International Conference on World Wide Web*, pages 297–306, New York, NY, USA, 2008. ACM.

[903] A. Kent, M. M. Berry, F. U. Luehrs Jr., and J. W. Perry. Operational criteria for designing information retrieval systems. *American Documentation*, 6(2), 1955.

[904] R. Khare and A. Rifkin. XML: A door to automated Web applications. *IEEE Internet Computing*, 1(4):78–86, 1977.

[905] R. Khare and A. Rifkin. The origin of (document) species. *Computer Networks and ISDN Systems*, 30(1-7), 1998. WWW7 Conference, Brisbane, Australia, available at http://decweb.ethz.ch/WWW7/00/.

[906] P. Kilpeläinen and H. Mannila. Retrieval from hierarchical texts by partial patterns. In *Proceedings of the 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Pittsburgh, PA, USA, 1993*, pages 214–222, 1993.

[907] B. J. Kim, A. Trusina, P. Minnhagen, and K. Sneppen. Self organized scale-free networks from merging and regeneration, Mar 2004.

[908] D. W. King and C. Tenopir. Evolving journal costs: Implications for publishers, libraries, and readers. *Learned Publishing*, 12:251–258, Oct. 1999.

[909] S. T. Kirsch. Document retrieval over networks wherein ranking and relevance scores are computed at the client for multiple database documents. US Patent 5,659,732, Aug. 1997.

[910] A. Kleiboemer, M. Lazear, and J. Pedersen. Tailoring a retrieval system for naive users. In *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval (SDAIR '96)*, Las Vegas, NV, 1996.

[911] J. Kleinberg. Authoritative sources in a hyperlinked environment. *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 46(5):604–632, 1998. http://www.cs.cornell.edu/home/kleinber/auth.pdf.

[912] D. Knuth, J. Morris, and V. Pratt. Fast pattern matching in strings. *SIAM J. Comput*, 6(2):323–350, June 1977.

[913] D. E. Knuth. *The Art of Computer Programming*, volume 3: Searching and Sorting. Addison-Wesley, 1973.

[914] T. Kochtanek and J. Matthews. *Library Information Systems: From Library Automation to Distributed Information.* Libraries Unlimited, 2002.

[915] T. R. Kochtanek and K. K. Hein. Delphi study of digital libraries. *Information Processing & Management*, 35(3):245–254, 1999.

[916] I. Kodratoff and J. Carbonell. *Machine Learning: An Artificial Intelligence Approach, Vol. III.* Kaufman Publishers Inc., 1990.

[917] W. Koehler. A longitudinal study of Web pages continued: a consideration of document persistence. *Information Research*, 9(2), January 2004.

[918] J. Koenemann and N. J. Belkin. A case for interaction: a study of interactive information retrieval behavior and effectiveness. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 205–212, 1996.

[919] R. Kohavi, R. Henne, and D. Sommerfield. Practical guide to controlled experiments on the web: listen to your customers not to the hippo. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'07)*, pages 959–967. ACM Press New York, NY, USA, 2007.

[920] R. Kohavi, R. M. Henne, and D. Sommerfield. Practical guide to controlled ex-

periments on the web: listen to your customers not to the hippo. In P. Berkhin, R. Caruana, and X. Wu, editors, *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA, August 12-15, 2007*, pages 959–967. ACM, 2007.

[921] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Min. Knowl. Discov.*, 18(1):140 181, 2009.

[922] R. Kohavi, L. Mason, R. Parekh, and Z. Zheng. Lessons and Challenges from Mining Retail E-Commerce Data. *Machine Learning*, 57(1):83–113, 2004.

[923] R. Kohavi, D. Sommerfield, and J. Dougherty. Data mining using MLC++, a machine learning library in C++. In *ICTAI '96: Proceedings of the 8th International Conference on Tools with Artificial Intelligence*, page 234, Washington, DC, USA, 1996. IEEE Computer Society.

[924] A. Kolcz and A. Chowdhury. Hardening fingerprinting by context. In *CEAS 2007 - The Fourth Conference on Email and Anti-Spam*, Mountain View, CA, USA, 2007.

[925] A. Kolcz and A. Chowdhury. Lexicon randomization for near-duplicate detection with I-Match. *The Journal of Supercomputing*, 45(3):255–276, 2008.

[926] A. Kolcz, A. Chowdhury, and J. Alspector. The impact of feature selection on signature-driven spam detection. In *CEAS 2004 - First Conference on Email and Anti-Spam*, Mountain View, CA, USA, July 2004.

[927] A. Kolcz, A. Chowdhury, and J. Alspector. Improved robustness of signature-based near-replica detection via lexicon randomization. In W. Kim, R. Kohavi, J. Gehrke, and W. DuMouchel, editors, *KDD*, pages 605–610, Seattle, WA, USA, August 2004. ACM.

[928] D. Konopnicki and O. Shmueli. W3QS: A query system for the World Wide Web. In *Proc. of VLDB'95*, pages 54–65, Zurich, Switzerland, Sept. 1995.

[929] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In *WWW 2009*, pages 171–180, 2009.

[930] J. Korpela. Lurching Toward Babel: HTML, CSS, and XML. *IEEE Computer*, 31(7):103–106, 1998.

[931] R. Korphage. *Information Storage and Retrieval.* John Wiley & Sons, Inc., 1997.

[932] J. Korst and V. Pronk. *Multimedia Storage and Retrieval: An Algorithmic Approach.* John Wiley & Sons, 2005.

[933] M. Koster. Guidelines for robots writers. http://www.robotstxt.org/wc/guidelines.html, 1993.

[934] M. Koster. Robots in the web: threat or treat ? *ConneXions*, 9(4), April 1995.

[935] M. Koster. A standard for robot exclusion. http://www.robotstxt.org/wc/exclusion.html, 1996.

[936] N. Koudas, C. Faloutsos, and I. Kamel. Declustering spatial databases on a multi-computer architecture. *EDBT Conf. Proc.*, pages 592–614, Mar. 1996.

[937] G. Kowalski and M. T. Maybury. *Information Storage and Retrieval Systems: Theory and Implementation.* Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[938] D. Kraft and D. Buel. Fuzzy sets and generalized Boolean retrieval systems. *International Journal of Man-Machine Studies*, 19:45–56, 1983.

[939] R. Krishnan. Google notebook blog. http://googlenotebookblog.blogspot.com/2009/01/stopping-development-on-google-notebook.html, Jan 2009.

[940] A. Krowne. Planetmath. http://planetmath.org/.

[941] A. Krumpholz and D. Hawking. InexBib - retrieving XML elements based on external evidence. *Australian Journal of Intelligent Information Processing Systems. ADCS 2006 special issue.*, 9(2):72–79, December 2006. http://es.csiro.au/pubs/krumpholz-hawking-adcs2006.pdf.

[942] U. Kruschwitz. *Intelligent Document Retrieval: Exploiting Markup Structure (The Information Retrieval Series).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[943] J. Kubica, A. Moore, D. Cohn, and J. Schneider. cgraph: A fast graph-based method for link analysis and queries, 2003.

[944] K. Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–440, Dec. 1992.

[945] B. Kules and B. Shneiderman. Users can change their Web search tactics: Design guidelines for categorized overviews. *Information Processing and Management*, 44(2):463–484, 2008.

[946] R. Kumar, J. Novak, B. Pang, and A. Tomkins. On anonymizing query logs via token-based hashing. In *WWW'07: Proceedings of the 16th international conference on World Wide Web*, pages 629–638, New York, NY, USA, 2007. ACM Press.

[947] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks*, 31(11–16):1481–1493, 1999.

[948] M. Kuniavsky. *Observing the User Experience: A Practitioner's Guide to User Research.* Morgan Kaufmann, 2003.

[949] K. Kwok. A neural network for probabilistic information retrieval. In *Proc. ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 21–30, 1989.

[950] K. Kwok. Experiments with a component theory of probabilistic information retrieval based on single terms as document components. *ACM Transactions on Information Systems*, 8(4):363–386, October 1990.

[951] K. Kwok. A network approach to probabilistic information retrieval. *ACM Transactions on Information Systems*, 13(3):324–353, July 1995.

[952] K. Kwok, L. Papadopolous, and Y. Kwan. Retrieval experiments with a large collection using pircs. In *Proc of the First TExt Retrieval Conference (TREC-1)*, USA, 1993. Special Publication 500-267, National Institute of Standards and Technology (NIST).

[953] K. L. Kwok. An attempt to identify weakest and strongest queries. In *Proceedings of the 28th Annual Conference on Research and Development in Information Retrieval (SIGIR)*, 2005.

[954] A. Lacerda, M. Cristo, M. A. Gonçalves, W. Fan, N. Ziviani, and B. A. Ribeiro-Neto. Learning to advertise. In *Proceedings of the 29th ACM Int. Conference on Information Retrieval, ACM SIGIR*, pages 549–556, 2006.

[955] A. H. F. Laender, B. A. Ribeiro-Neto, and A. S. da Silva. DEByE - data extraction by example. *Data and Knowledge Engineering*, 40(2):121–154, 2002.

[956] B. Lagoeiro, M. A. Gonçalves, and A. H. F. Laender. 5SQual - A Quality Assessment Tool for Digital Libraries. In *JCDL '07: Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, page (accepted for publication), 2007.

[957] C. Lagoze. Networked Computer Science Technical Reference Library. http://www.ncstrl.org.

[958] C. Lagoze. The Warwick framework: A container architecture for diverse sets of metadata. *D-Lib Magazine*, 2(7), July 1996.

[959] C. Lagoze, W. Arms, S. Gan, D. Hillmann, C. Ingram, D. Krafft, R. Marisa, J. Phipps, J. Saylor, C. Terrizzi, W. Hoehn, D. Millman, J. Allan, S. Guzman-Lara, and T. Kalt. Core services in the architecture of the national science digital library (NSDL). In *Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 201–209, Portland, Oregon, 2002.

[960] C. Lagoze, W. Y. Arms, S. Gan, D. Hillmann, C. Ingram, D. B. Krafft, R. J. Marisa, J. Phipps, J. Saylor, C. Terrizzi, W. Hoehn, D. Millman, J. Allan, S. Guzman-Lara, and T. Kalt. Core services in the architecture of the national science digital library (nsdl). In *ACM/IEEE Joint Conference on Digital Libraries (JCDL 2002)*, pages 201–209, Portland, Oregon, 2002.

[961] C. Lagoze, D. Fielding, and S. Payette. Making global digital libraries work: Collection services, connectivity regions, and collection views. In I. Witten, R. Akscyn, and F. M. Shipman, editors, *Proc. of the 3rd ACM Conf. on Digital Libraries (DL-98)*, pages 134–143, jun 1998.

[962] C. Lagoze, S. Payette, E. Shin, and C. Wilper. Fedora: an architecture for complex objects and their relationships. *Int. J. on Digital Libraries*, 6(2):124–138, 2006.

[963] C. Lagoze and H. van de Sompel. The Open Archives Initiative. In *Proc. of the 1st Joint Conf. on Digital Libraries (JCDL'2001)*, pages 54–62, Roanoke, Virginia, June 24-28, 2001.

[964] L. V. S. Lakshmanan, F. Sadri, and I. N. Subramanian. A declarative language for querying and restructuring the Web. In *Proc. of 6th. International Workshop on Research Issues in Data Engineering, RIDE '96*, New Orleans, Feb. 1996.

[965] M. Lalmas. *XML Retrieval.* Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan Claypool, 2009.

[966] M. Lalmas and V. Murdock, editors. *ACM SIGIR Workshop on Aggregated Search*, Singapore, 2008.

[967] M. Lalmas and I. Ruthven. Representing and retrieving structured documents using the dempster-shafer theory of evidence: Modelling and evaluation. *Journal of Documentation*, 54(5):529–565, 1998.

[968] M. Lalmas and A. Tombros. Evaluating XML Retrieval Effectiveness at INEX. *SIGIR Forum*, 41(1):40–57, 2007.

[969] Lam, Wai and Lai, Kwok-Yin. A meta-learning approach for text categorization. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 303–309, New Orleans, Lousiana, 2001.

[970] B. LaMacchia. The Internet fish construction kit. In *6th. Int'l. WWW Conference*, Santa Clara, CA, USA, Apr. 1997.

[971] L. Lamport. Paxos made simple. *ACM SIGACT News*, 32(4):51–58, December 2001.

[972] F. Lancaster. *Indexing and Abstracting in Theory and Practice.* University of Illinois, 3rd edition, 2003.

[973] G. Landau and U. Vishkin. Fast string matching with *k* differences. *Journal of Computer Systems Science*, 37:63–78, 1988.

[974] T. Landauer, D. Egan, J. Remde, M. Lesk, C. Lochbaum, and D. Ketchum. Enhancing the usability of text through computer delivery and formative evaluation: the superbook project. In C. McKnight, A. Dillon, and J. Richardson, editors, *Hypertext: A Psychological Perspective*, pages 71–136. Ellis Horwood, 1993.

[975] P. Langley. *Elements of Machine Learning.* Morgan Kaufmann, 1996.

[976] A. N. Langville and C. D. Meyer. *Google's PageRank and Beyond: The Science*

of Search Engine Rankings. Princeton University Press, July 2006.

[977] A. Large, L. Tedd, and R. Hartley, editors. *Information Seeking in The Online Age: Principles and Practice.* Bowker-Saur, London, UK, 1999.

[978] L. S. Larkey, M. E. Connell, and J. Callan. Collection selection and results merging with topically organized u.s. patents and trec data. In *CIKM '00: Proceedings of the ninth international conference on Information and knowledge management*, pages 282–289, New York, NY, USA, 2000. ACM Press.

[979] L. S. Larkey and W. B. Croft. Combining classifiers in text categorization. In H.-P. Frei, D. Harman, P. Schäuble, and R. Wilkinson, editors, *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*, pages 289–297, Zürich, CH, 1996. ACM Press, New York. US.

[980] R. R. Larson. Cheshire II at INEX'03: Component and Algorithm Fusion for XML Retrieval. In *INEX 2003 Proceedings*, pages 38–45, 2003.

[981] N. Larsson and A. Moffat. Offline dictionary-based compression. *Proceedings of the IEEE*, 88(11):1722 1732, 2000.

[982] O. Lassila. Web metadata: A matter of semantics. *IEEE Internet Computing*, 2(4):30–37, 1998.

[983] O. Lassila and R. Swick. World Wide Web Consortium - RDF. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/, 1999.

[984] E. Lau and D. Goh. In search of query patterns: a case study of a university OPAC. *Information Processing and Management*, 42(5):1316–1329, 2006.

[985] V. Lavrenko and W. B. Croft. Relevance based language models. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 120–127, 2001.

[986] S. Lawrence and C. L. Giles. Context and page analysis for improved Web search. *IEEE Internet Computing*, 2(4):38–46, 1998.

[987] S. Lawrence and C. L. Giles. Inquirus, the NECI meta search engine. In *7th WWW Conference*, pages 95–105, Brisbane, Australia, 1998.

[988] S. Lawrence, C. L. Giles, and K. Bollacker. Digital libraries and autonomous citation indexing. *Computer*, 32(6):67–71, 1999.

[989] S. Lawrence and L. C. Giles. Accessibility of information on the web. *Intelligence*, 11(1):32–39, 2000.

[990] A. Lazonder, H. Biemans, and I. Wopereis. Differences Between Novice and Experienced Users in Searching Information on the World Wide Web. *Journal of the American Society for Information Science*, 51(6):576–581, 2000.

[991] D. Lea. *Concurrent Programming in Java: Design Principles and Patterns.* The Java Series. Addison-Wesley, Reading, MA, 1997.

[992] C. P. Lee, G. H. Golub, and S. A. Zenios. A fast two-stage algorithm for computing pagerank and its extensions. Technical report, Stanford University, 2004.

[993] J. Lee and P. Kantor. A study of probabilistic information retrieval systems in the case of inconsistent expert judgements. *Journal of the American Society for Information Sciences*, 42(3):166-172, 1991.

[994] J. Lee, W. Kim, and Y. Lee. Ranking documents in thesaurus-based Boolean retrieval systems. *Information Processing & Management (IP&M)*, 30(1):79–91, 1993.

[995] J. H. Lee. Properties of extended Boolean models in information retrieval. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Statistical Models, pages 182–190, 1994.

[996] J. H. Lee, W. Y. Kim, M. H. Kim, and Y. J. Lee. On the evaluation of Boolean operators in the extended Boolean retrieval framework. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Mathematical Models, pages 291–297, 1993.

[997] U. Lee, Z. Liu, and J. Cho. Automatic identification of user goals in Web search. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 391–400, New York, NY, USA, 2005. ACM Press.

[998] Y.-B. Lee and S. H. Myaeng. Text genre classification with genre-revealing and subject-revealing features. In *Proceedings of SIGIR-02, 25th ACM International Conference on Research and Development in Information Retrieval*, pages 145–150, Tampere, FI, 2002.

[999] W. G. LeFurgy. Building preservation partnerships: the library of congress national digital information infrastructure and preservation program. *Library Trends*, 54(1), 2005. http://www.digitalpreservation.gov/library/pdf/building.pdf.

[1000] W. G. LeFurgy, M. Hedstrom, T. A. Pardo, and T. O. Walters. Preserving information long-term: digital archiving. In L. M. L. Delcambre and G. Giuliano, editors, *Proceedings of the 2005 National Conference on Digital Government Research, DG.O 2005, Atlanta, Georgia, USA, May 15-18, 2005*, page 15. Digital Government Research Center, 2005.

[1001] J. Lehman. Building a taxonomy. Technical Report 2, New Idea Engineering - NIE Enterprise Search, june 2003. http://www.ideaeng.com/pub/entsrch/issue02/article02.html.

[1002] R. Lempel and S. Moran. Predictive caching and prefetching of query results in search engines. In *WWW'03: Proceedings of the 12th international conference on World Wide Web*, pages 19-28, New York, NY, USA, 2003. ACM Press.

[1003] Lemur toolkit. http://www.lemurproject.org/, 2007.

[1004] M. Lesk. Word-word associations in document retrieval systems. *American Documentation*, 20(1):8-36, 1969.

[1005] M. Lesk. *Practical Digital Libraries; Books, Bytes, & Bucks*. Morgan Kaufman, 1997.

[1006] M. Lesk. *Understanding Digital Libraries*. Morgan Kaufmann, 2nd edition, 2005.

[1007] J. Leskovec, S. Dumais, and E. Horvitz. Web Projections: Learning from Contextual Subgraphs of the Web. In *Int'l Conference of the World Wide Web*, 2007.

[1008] O. Levard. Google peut-il vous traiter d'arnaqueur? *LCI*, July 21 2009. http://tf1.lci.fr/infos/economie/entreprises/0,4490540,00.html (in French).

[1009] O. Levard. Les suggestions très limites de google. *LCI*, July 16 2009. http://tf1.lci.fr/infos/high-tech/0,4227853,00-les-suggestions-tres-limites-de-google-.html (in French).

[1010] M. Levene and A. Poulovassilis. *Web Dynamics*. Springer, 2004.

[1011] M. Levene and A. Poulovassilis. Special issue on Web dynamics. *Computer Networks*, 50(10):1425–1429, 2006.

[1012] V. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Phys. Dokl*, 6:126–136, 1966.

[1013] D. M. Levy. Heroic measures: reflections on the possibility and purpose of digital preservation. In *DL'98: Proceedings of the 3rd ACM International Conference on Digital Libraries*, pages 152–161, Pittsburgh, PA, 1998.

[1014] D. D. Lewis. Naive (Bayes) at forty: the independence assumption in information retrieval. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 4–15, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[1015] D. D. Lewis and M. Ringuette. A comparison of two learning algorithms for text categorization. In *Symposium on Document Analysis and Information Retrieval*, pages 81–93, Las Vegas, Nevada, Apr. 1994.

[1016] D. D. Lewis, Y. Yang, T. G. Rose, G. Dietterich, F. Li, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[1017] LexisNexis. Data Centres. http://www.lexisnexis.com/presscenter/mediakit/datacenter.asp.

[1018] M. Li, M. Zhu, Y. Zhang, and M. Zhou. Exploring Distributional Similarity Based Models for Query Spelling Correction. In *Annual Meeting-Association for Computational Linguistics (ACL'06)*, 2006.

[1019] P. Li, C. J. C. Burges, and Q. Wu. Mcrank: Learning to rank using multiple classification and gradient boosting. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *NIPS*. MIT Press, 2007.

[1020] W.-S. Li, J. Shim, K. Candan, and Y. Hara. WebDB: A Web query system and its modeling, language, and implementation. In *Proc. of Advances in Digital Libraries*, Santa Barbara, CA, USA, April 1998.

[1021] Y. Li. Toward a qualitative search engine. *IEEE Internet Computing*, 2(4):24–29, July 1998.

[1022] Y. H. Li and A. K. Jain. Classification of text documents. *Comput. J*, 41(8):537–546, 1998.

[1023] Library of Congress, Z39.50 Maintenance Agency, June 1998. http://lcweb.loc.gov/z3950/agency/.

[1024] Library of Congress. The national digital information infrastructure and preservation program, 2006. http://www.digitalpreservation.gov/.

[1025] Library of Congress, Metadata Encoding and Transmission Standard (METS), May 2006. http://www.loc.gov/standards/mets/.

[1026] Library of Congress, Network Development and MARC Standards Office. Metadata object description schema (MODS) version 3.1, July 2005.

[1027] H. Lie and B. Bos. *Cascading Style Sheets: Designing for the Web*. Addison-Wesley, 1997.

[1028] R. Lienhart. Comparison of automatic shot boundary detection algorithms. *SPIE Image and Video Processing VII*, pages 3656–3729, January 1999.

[1029] R. Lienhart. Reliable transition detection in videos: a survey and practitioner' s guide. *International Journal of Image and Graphics*, 1(3):469–486, 2001.

[1030] R. Lienhart, S. Pfeiffer, and W. Effelsberg. The MoCA Workbench: Support for creativity in movie content analysis. In *ICMCS*, pages 314–321, 1996.

[1031] L. Lim, M. Wang, S. Padmanabhan, J. S. Vitter, and R. Agarwal. Characterizing Web document change. In *Proceedings of the Second International Conference on Advances in Web-Age Information Management*, volume 2118 of *Lecture Notes in Computer Science*, pages 133–144, London, UK, July 2001. Springer.

[1032] L. R. S. Lima, A. H. F. Laender, and B. A. A. Ribeiro-Neto. A hierarchical approach to the automatic categorization of medical documents. In *CIKM '98: Proceedings of the seventh international conference on Information and knowledge management*, pages 132–139, New York, NY, USA, 1998. ACM.

[1033] J. Lin, M. DiCuccio, V. Grigoryan, and W. Wilbur. Navigating information spaces: A case study of related article search in PubMed. *Information Processing and Management*, 2008.

[1034] J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. Karger. What Makes a Good Answer? the Role of Context in Question Answering. *Proceedings of Human-Computer Interaction (INTERACT'03)*, 2003.

[1035] K.-I. Lin, H. Jagadish, and C. Faloutsos. The TV-tree - an index structure for high-dimensional data. *VLDB Journal*, 3:517–542, Oct. 1994.

[1036] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[1037] B. Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric Systems and Applications)*. Springer, 1st ed. 2007. corr. 2nd printing edition, January 2009.

[1038] F. Liu and R. Picard. Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(7):722–733, July 1996.

[1039] T.-Y. Liu, J. Xu, T. Qin, W. Xiong, and H. Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *LR4IR Workshop, in conjunction with SIGIR 2007*, 2007.

[1040] T.-Y. Liu, Y. Yang, H. Wan, H.-J. Zeng, Z. Chen, and W.-Y. Ma. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explorations*, 7(1):36–43, 2005.

[1041] X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 186–193, New York. NY. USA. 2004. ACM Press.

[1042] X. Liu and W. B. Croft. Statistical language modeling for information retrieval. In B. Cronin, editor, *Annual Review of Information Science and Technology*, volume 39. ASIS&T, 2005. Chapter 1.

[1043] Y. Liu, M. Zhang, and R. Cen. Data cleansing for Web information retrieval using query independent features. *Journal of the American Society for Information Science and Technology*, 58(12):001–015, 2007.

[1044] M.-L. Lo and C. V. Ravishankar. Spatial joins using seeded trees. In *Proc. of ACM SIGMOD*, pages 209–220, Minneapolis, MN, USA, May 1994.

[1045] Load monitor project. http://sourceforge.net/projects/monitor, 2007.

[1046] X. Long and T. Suel. Optimized query execution in large search engines with global page ordering. In *Proceedings of VLDB 2003*, pages 129–140, 2003.

[1047] X. Long and T. Suel. Three-Level Caching for Efficient Query Processing in Large Web Search Engines. In *WWW'05: Proceedings of the 14th International World Wide Web conference*, Chiba, Japan, 2005.

[1048] B. T. Loo, R. Huebsch, J. M. Hellerstein, S. Shenker, and I. Stoica. Enhancing P2P File-Sharing with an Internet-Scale Query Processor. In *VLDB'04: Proceedings of the 30th International conference on Very Large Data Bases*, Toronto, Canada, 2004.

[1049] R. A. Lorie. Long term preservation of digital information. In *Proceedings of the 1st ACM/IEEE Joint Conference on Digital Libraries*, pages 346–352, Roanakoe, Virginia, 2001.

[1050] R. Losee and A. Bookstein. Integrating Boolean queries in conjunctive normal form with probabilistic retrieval models. *Information Processing & Management (IP&M)*, 24(3):315–321, 1988.

[1051] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[1052] R. Lowry. *Concepts and Applications of Inferential Statistics.* Vassar College, Poughkeepsie, NY, USA, 2008. http://faculty.vassar.edu/lowry/webtext.html.

[1053] J. Lu and J. Callan. Content-Based Retrieval in Hybrid Peer-to-Peer Networks. In *Proc. of ACM Int'l Conf. on Information and Knowledge Management*, pages 199–206, 2003.

[1054] J. Lu and J. Callan. Federated Search of Text-Based Digital Libraries in Hierarchical Peer-to-Peer Networks. In *ECIR'05: Proceedings of the 27th European conference on IR Research*, Santiago de Compostela, Spain, 2005.

[1055] J. Lu and J. Callan. User Modeling for Full-Text Federated Search in Peer-to-Peer Networks. In *SIGIR'06: Proceedings of the 29th International ACM SIGIR conference on Research and Development in Information Retrieval*, Seattle, WA, USA, 2006.

[1056] Q. Lu, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *ICS '02: Proceedings of the 16th international conference on Supercomputing*, pages 84 95, New York, NY, USA, 2002. ACM.

[1057] W. Lu, S. Robertson, and A. MacFarlane. Field-Weighted XML Retrieval Based on BM25. In *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005*, pages 161–171, Dagstuhl Castle, Germany, 2005. Revised Selected Papers.

[1058] Y. Lu, C. Hu, X. Zhu, H. Zhang, and Q. Yang. A unified framework for semantics and feature based relevance feedback in image retrieval systems. In *Proceedings of the eighth ACM international conference on Multimedia*, pages 31–37, 2000.

[1059] Z. Lu, K. S. McKinley, and B. Cahoon. The hardware/software balancing act for information retrieval on symmetric multiprocessors. Technical Report TR98-25, Dept. of Comp. Sci., Univ. of Mass., Amherst, MA, 1998.

[1060] C. Lucchese, S. Orlando, R. Perego, and F. Silvestri. Mining query logs to optimize index partitioning in parallel Web search engines. In *Proceedings of the 2nd international conference on Scalable information systems*, Suzhou, China, 2007.

[1061] Lucene. http://jakarta.apache.org/lucene/, 2007.

[1062] H. Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 1(4):309–317, 1957.

[1063] H. Luhn. *Keyword-in-context Index for Technical Literature (KWIC Index).* International Business Machines Corp., Advanced Systems Development Division, 1959.

[1064] T.-Y. Lui. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[1065] R. P. Luk, H. V. Leong, T. Dillon, A. S. Chan, W. B. Croft, and J. Allan. A survey in indexing and searching XML documents. *Journal of the American Society for Information Science and Technology*, 53(6):415–437, 2002.

[1066] C. A. Lynch and J. K. Lippincott. Institutional Repository Deployment in the United States as of Early 2005. *D-Lib Magazine*, 11, 2005. http://www.dlib.org/dlib/september05/lynch/09lynch.html.

[1067] Y. Maarek, M. Jacovi, M. Shtalhaim, S. Ur, D. Zernik, and I. Ben-Shaul. Webcutter: a system for dynamic and tailorable site mapping. In *Selected papers from the sixth international conference on World Wide Web*, pages 1269–1279, Essex, UK, 1997. Elsevier Science Publishers Ltd.

[1068] Y. Maarek and D. Zernick. Proceedings of the WWW6 Workshop on Site Mapping, April 1997.

[1069] D. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Sept. 2003.

[1070] I. Macleod. Storage and retrieval of structured documents. *Information Processing & Management*, 26(2):197–208, 1990.

[1071] I. MacLeod. A query language for retrieving information from hierarchic text structures. *The Computer Journal*, 34(3):254–264, 1991.

[1072] I. A. Macleod, T. P. Martin, B. Nordin, and J. R. Phillips. Strategies for building distributed information retrieval systems. *Inf. Process. & Mgmnt.*, 23(6):511–528, 1987.

[1073] V. Mäkinen and G. Navarro. Succinct suffix arrays based on run-length encoding. *Nordic Journal of Computing*, 12(1):40–66, 2005.

[1074] Managing Gigabytes. http://www.cs.mu.oz.au/mg/, 2007.

[1075] U. Manber and G. Myers. Suffix arrays: a new method for on-line string searches. In *Proc. of ACM-SIAM Symposium on Discrete Algorithms*, pages 319–327, San Francisco, USA, 1990.

[1076] U. Manber, A. Patel, and J. Robison. The business of personalization: Experience with personalization of Yahoo! *Commun. ACM*, 43(8):35–39, 2000.

[1077] U. Manber, M. Smith, and B. Gopal. WebGlimpse: combining browsing and searching. In *Proc. of USENIX Technical Conference*, pages 195–206, Anaheim, USA, Jan 1997.

[1078] U. Manber and S. Wu. GLIMPSE: A tool to search through entire file systems. In *Proceedings of the Winter 1994 USENIX Conference: January 17-21, 1994, San Francisco, California, USA*, pages 23–32, Berkeley, CA, USA, Winter 1994.

[1079] P. Maniatis, M. Roussopoulos, T. J. Giuli, D. S. H. Rosenthal, and M. Baker. The LOCKSS peer-to-peer digital preservation system. *ACM Transactions on Computer Systems*, 23(1):2–50, 2005.

[1080] B. Manjunat, P. Salembier, and T. Sikora. *Introduction to MPEG-7: Multimedia Content Description Interface*. Wiley and Sons, 2002.

[1081] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[1082] G. Marchionini. *Information Seeking in Electronic Environments*. Cambridge University Press, 1995.

[1083] G. Marchionini. Evaluating digital libraries: A longitudinal and multifaceted view. *Library Trends*, 49(2), 2000.

[1084] G. Marchionini. A briefing on the evolution and status of the open video digital library. *Int. J. on Digital Libraries*, 4(1):36–38, 2004.

[1085] G. Marchionini. Exploratory Search: From Finding To Understanding. *Communications of the Acm*, 49(4):41–49, 2006.

[1086] M. Marchiori. The quest for correct information of the Web: hyper search engines. In *Proc. of the sixth international conference on the Web*, pages 265–274, Santa Clara, CA, USA, April 1997.

[1087] M. Marín, C. Bonacic, V. G. Costa, and C. Gómez. A Search Engine Accepting On-Line Updates. In *13th European International Conference on Parallel Processing (Euro-Par 2007)*, LNCS 4641, pages 348–357, Rennes, France, 2007.

[1088] M. Marín and V. G. Costa. High-performance Distributed Inverted Files. In

ACM 16th Conference on Information and Knowledge Management (CIKM 2007), pages 935–938, Lisbon, Portugal, Nov. 2007.

[1089] M. Marín and V. G. Costa. (Sync|Async)$^+$ MPI Search Engines. In *14th European PVM/MPI Meeting*, LNCS 4757, pages 117–124, Paris, 2007.

[1090] M. Marín and C. Gómez. Load Balancing Distributed Inverted Files. In *9th ACM International Workshop on Web Information and Data Management (WIDM 2007)*, pages 57–64, Lisbon, Portugal, November 2007.

[1091] E. P. Markatos. On caching search engine query results. *Computer Communications*, 24(2):137–143, 2001.

[1092] J. Markwell and D. W. Brooks. Link-rot limits the usefulness of Web-based educational materials in biochemistry and molecular biology. *Biochem. Mol. Biol. Educ.*, 31:69–72, 2003.

[1093] M. Maron and J. Kuhns. On relevance, probabilistic indexing and information retrieval. *Journal of ACM*, 7(3):216–244, 1960.

[1094] C. C. Marshall. Making metadata: A study of metadata creation for a mixed physical-digital collection. In *DL'98: Proceedings of the 3rd ACM International Conference on Digital Libraries*, pages 162–171, 1998.

[1095] B. Masand, G. Linoff, and D. Waltz. Classifying news stories using memory based reasoning. In N. Belkin, P. Ingwersen, and A. M. Pejtersen, editors, *Proceedings of the 15th Annual International Conference on Reasearch and Development in Information Retrieval*, SIGIR Forum, pages 59–65, New York, NY, USA, June 1992. ACM Press.

[1096] M. Masnick. Two separate rulings in france split over whether google's suggestion algorithm can be libelous. *Techdirt*, July 24 2009. http://www.techdirt.com/articles/20090724/0407145647.shtml.

[1097] Y. Mass and M. Mandelbrod. Retrieving the most relevant XML Components. In *INEX 2003 Proceedings*, pages 53–58, 2003.

[1098] Y. Mass and M. Mandelbrod. Component Ranking and Automatic Query Refinement for XML Retrieval. In *Advances in XML Information Retrieval, Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*, pages 73–84, Dagstuhl Castle, Germany, 2005. Revised Selected Papers.

[1099] Y. Mass and M. Mandelbrod. Using the INEX Environment as a Test Bed for Various User Models for XML Retrieval. In *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005*, pages 187–195, Dagstuhl Castle, Germany, 2006. Revised Selected Papers.

[1100] M. Massey and W. Bender. Salient stills: process and practice. *IBM Syst. J.*, 35(3-4):557–573, 1996.

[1101] M. T. Maybury. *Intelligent Multimedia Information Retrieval.* MIT Press, 1997.

[1102] M. Mayer. Universal search: the best answer is still the best answer. *The Official Google Blog*, May 2007. http://googleblog.blogspot.com/2007/05/universal-search-best-answer-is-still.html.

[1103] O. A. Mcbryan. GENVL and WWWW: Tools for taming the web. In *Proceedings of the first World Wide Web Conference*, Geneva, Switzerland, May 1994.

[1104] A. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/~mccallum/bow, 1996.

[1105] A. McCallum and K. Nigam. A comparison of event models for naive Bayes text classification. In *AAAI/ICML-98 Workshop on Learning for Text Categorization*, pages 41–48. AAAI Press, 1998.

[1106] J. McCarthy. Artificial intelligence, logic and formalizing common sense. In R. Thomason, editor, *Philosophical Logic and Artificial Intelligence*. Kluver Academic, 1989.

[1107] J. McCarthy. *Formalizing Common Sense: Papers by John McCarthy*. Ablex Publishing Corporation, 1990.

[1108] E. McCreight. A space-economical suffix tree construction algorithm. *Journal of the ACM*, 23(2):262–272, 1976.

[1109] K. McKeown, J. Hirschberg, M. Galley, and S. Maskey. From text to speech summarization. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages v/997–v1000, March 2005.

[1110] F. McMartin and Y. Terada. Digital library services for authors of learning materials. In *Proc. of JCDL'02*, pages 117–118, Portland, OR, 2002.

[1111] F. Mcsherry. A uniform approach to accelerated pagerank computation. In *WWW'05: Proceedings of the 14th international conference on World Wide Web*, pages 575–582, New York, NY, USA, 2005. ACM Press.

[1112] C. T. Meadow, B. R. Boyce, D. H. Kraft, and C. L. Barry. *Text Information Retrieval Systems, Third Edition*. Academic Press, Inc., Orlando, FL, USA, 2007.

[1113] M. Mealling and R. Denenberg. Uniform resource identifiers (URIs), URLs, and uniform resource names (URNs): Clarifications and recommendations. Internet informational RFC 3305, Aug. 2002.

[1114] R. Meddis and M. J. Hewitt. Virtual pitch and phase sensitivity of a computer model of the auditory periphery. I: Pitch identification, and II: Phase sensitivity. *J. Acoust. Soc. Am.*, 89:2866–2894, 1991.

[1115] MEDLINE. NLM—United States National Library of Medicine, 2009.

[1116] C. Meilhac and C. Nastar. Relevance feedback and category search in image databases. In *IEEE International Conference on Multimedia Computing*, 1999.

[1117] S. Melnik, S. Raghavan, B. Yang, and H. Garcia-Molina. Building a distributed full-text index for the web. *ACM Trans. Inf. Syst.*, 19(3):217–241, 2001.

[1118] D. A. Menascé and V. A. Almeida. *Capacity Planning for Web Performance: Metrics, Models, and Methods*. Prentice Hall, 1998.

[1119] F. Menczer. Lexical and semantic clustering by Web links. *Journal of the American Society for Information Science and Technology*, 55(14):1261–1269, August 2004.

[1120] A. Mendelzon, G. Mihaila, and T. Milo. Querying the World Wide Web. *International Journal on Digital Libraries*, 1(1):54–67, April 1997.

[1121] D. Merrill. http://www.youtube.com/watch?v=syKY8CrHkck\#t=22m11sattimestamp22m11s.

[1122] R. Michalski, J. Carbonell, and T. Mitchell. *Machine Learning: An Artificial Intelligence Approach, Vol. I*. Kaufman Publishers Inc., 1983.

[1123] R. Michalski, J. Carbonell, and T. Mitchell. *Machine Learning: An Artificial Intelligence Approach, Vol. II*. Kaufman Publishers Inc., 1986.

[1124] S. Michel, M. Bender, N. Ntarmos, P. Triantafillou, G. Weikum, and C. Zimmer. Discovering and Exploiting Keyword and Attribute-Value Co-occurrences to Improve P2P Routing Indices. In *CIKM'06: Proceedings of the 15th ACM International conference on Information and Knowledge Management*, Arlington, Virginia, USA, 2006.

[1125] S. Michel, P. Triantafillou, and G. Weikum. KLEE: a framework for distributed

top-k query algorithms. In *VLDB'05: Proceedings of the 31st International conference on Very Large Data Bases*, Trondheim, Norway, 2005.

[1126] S. Michel, P. Triantafillou, and G. Weikum. MINERVA∞: A Scalable Efficient Peer-to-Peer Search Engine. In *Middleware'05: Proceedings of the 6th International Middleware conference*, Grenoble, France, 2005.

[1127] V. Mihajlovic, G. Ramírez, T. Westerveld, D. Hiemstra, H. E. Blok, and A. de Vries. TIJAH Scratches INEX 2005: Vague Element Selection, Image Search, Overlap, and Relevance Feedback. In *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005*, pages 72–87, Dagstuhl Castle, Germany, 2006. Revised Selected Papers.

[1128] P. Mika. *Social Networks and the Semantic Web*, volume 5 of *Semantic Web and Beyond*. Springer-Verlag, Berlin-Heidelberg, 2007.

[1129] P. Mika. Microsearch: An interface for semantic search. In *Proceedings of the SemSearch 2008 Workshop on Semantic Search at the 5th European Semantic Web Conference*, Tenerife, Spain, June 2008. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-334/.

[1130] N. Milic-Frayling, R. Jones, K. Rodden, G. Smyth, A. Blackwell, and R. Sommerer. Smartback: supporting users in back navigation. *Proceedings of the 13th International Conference on World Wide Web (WWW'04)*, pages 63–71, 2004.

[1131] D. R. Millen, J. Feinberg, and B. Kerr. Dogear: Social bookmarking in the enterprise. In *Proceedings of CHI '06*, pages 111–120, New York, NY, USA, 2006. ACM Press.

[1132] D. Miller, T. Leek, and R. Schwartz. A hidden Markov model information retrieval system. In *ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 214–221, 1999.

[1133] G. Miller, E. Newman, and E. Friedman. Length-frequency statistics for written english. *Information and Control*, 1:370–389, 1958.

[1134] R. Miller. Websphinx, a personal, customizable Web crawler. http://www-2.cs.cmu.edu/~rcm/websphinx, 2004.

[1135] R. Miller and K. Bharat. Sphinx: A framework for creating personal, site-specific Web crawlers. In *Proceedings of the seventh conference on World Wide Web*, Brisbane, Australia, April 1998. Elsevier Science.

[1136] M. Mills, J. Cohen, and Y. Y. Wong. A magnifier tool for video data. In *CHI '92: Proceedings of the SIGCHI Conference on Human factors in Computing Systems*, pages 93–98, New York, NY, USA, 1992. ACM.

[1137] J. Minker, G. Wilson, and B. Zimmerman. An evaluation of query expansion by the addition of clustered terms for a document retrieval system. *Information Storage and Retrieval*, 8(6):329–348, 1972.

[1138] T. Minohara and R. Watanabe. Queries on structure in hypertext. In *Foundations of Data Organization and Algorithms, FODO '93*, pages 394–411. Springer, 1993.

[1139] R. Mitchell, D. Day, and L. Hirschman. Fishing for information on the Internet. In *Proceedings '95 Information Visualization*, pages 105–111, Atlanta, USA, Oct. 1995.

[1140] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[1141] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In B. Croft, A. Moffat, C. van Rijsbergen, R. Wilkinson, and J. Zobel, editors, *Proc. of 21st Annual International Conference on Research and Development in Information Retrieval, SIGIR 98*, pages 206–214, Melbourne, Australia, 1998.

[1142] M. Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1(2):226–251, 2004.

[1143] S. Miyamoto, T. Miyake, and K. Nakayama. Generation of a pseudothesaurus for information retrieval based on cooccurrences and fuzzy set operations. *IEEE Transactions on Systems and Man Cybernetics*, 13(1):62–70, 1983.

[1144] S. Miyamoto and K. Nakayama. Fuzzy information retrieval based on a fuzzy pseudothesaurus. *IEEE Transactions on Systems and Man Cybernetics*, 16(2):278–282, 1986.

[1145] S. Mizzaro. Relevance: the whole history. *Journal of the American Society for Information Science*, 48(9):810–832, 1997.

[1146] W. E. Moen. *The development of ANSI/NISO Z39.50: A case study in standards evolution.* PhD thesis, Syracuse University, 1998.

[1147] A. Moffat. Word-based text compression. *Software Practice and Experience*, 19(2):185–198, 1989.

[1148] A. Moffat. *Compression and Coding Algorithms.* Kluwer, 2002.

[1149] A. Moffat and T. Bell. In situ generation of compressed inverted files. *Journal of the American Society for Information Science*, 46(7):537–550, 1995.

[1150] A. Moffat and R. Wan. Re-Store: A system for compressing, browsing, and searching large documents. In *Proc. 8th International Symposium on String Processing and Information Retrieval*, pages 162–174, 2001.

[1151] A. Moffat, W. Webber, and J. Zobel. Load balancing for term-distributed parallel retrieval. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 348–355, New York, NY, USA, 2006. ACM Press.

[1152] A. Moffat and J. Zobel. Information retrieval systems for large document collections. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, pages 85–94, Gaithersburg, MD, USA, 1995. Dept. of Commerce, National Institute of Standards and Technology. Special Publication 500-226.

[1153] A. Moffat and J. Zobel. What does it mean to "measure performance"? In X. Zhou, S. Su, M. P. Papazoglou, M. E. Owlowska, and K. Jeffrey, editors, *Proc. Fifth International Conf. on Web Informations Systems*, pages 1–12, Brisbane, Australia, Nov. 2004. LNCS 3306, Springer.

[1154] K. Monostori, R. A. Finkel, A. B. Zaslavsky, G. Hodász, and M. Pataki. Comparison of overlap detection techniques. In P. M. A. Sloot, C. J. K. Tan, J. Dongarra, and A. G. Hoekstra, editors, *International Conference on Computational Science (I)*, volume 2329 of *Lecture Notes in Computer Science*, pages 51–60, Amsterdam, The Netherlands, 2002. Springer.

[1155] K. Monostori, A. B. Zaslavsky, and H. W. Schmidt. Efficiency of data structures for detecting overlaps in digital documents. In *24th Australasian Computer Science Conference (ACSC 2001)*, pages 140–147, Gold Coast, QU, Australia, February 2001. IEEE Computer Society.

[1156] M. R. Morris and J. Teevan. *Collaborative Web Search: Who, What, Where, When, and Why.* Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan Claypool, 2009.

[1157] P. Morville and L. Rosenfeld. *Information Architecture for the World Wide Web: Designing Large-Scale Web Sites.* O'Reilly Media, 3rd edition, December 2006.

[1158] E. Moura, G. Navarro, N. Ziviani, and R. Baeza-Yates. Fast and flexible word searching on compressed text. *ACM Transactions on Information Systems (TOIS)*, 18(2):113–139, 2000.

[1159] M. A. Moura. Personal communication at the Information Sciences School, UFMG, Brazil, 2004.

[1160] F. Mourão, L. C. da Rocha, R. B. Araújo, T. Couto, M. A. Gonçalves, and W. Meira Jr. Understanding temporal aspects in document classification. In *Proceedings of the International Conference on Web Search and Web Data Mining, WSDM 2008, Palo Alto, California, USA, February 11-12, 2008*, pages 159–170, 2008.

[1161] S. Mukherjea and J. Foley. Visualizing the World Wide Web with the Navigational View Builder. *Computer Networks and ISDN Systems*, 27:1075–1087, 1995.

[1162] R. Mukherjee and J. Mao. Enterprise search: Tough stuff. *Queue*, 2(2):36–46, 2004.

[1163] H. Müller, W. Müller, D. Squire, S. Marchand-Maillet, and T. Pun. Performance evaluation in content-based image retrieval: overview and proposals. *Pattern Recognition Letters*, 22(5):593–601, 2001.

[1164] S. A. Murray. *The Library – An Illustrated History.* Skyhorse Publishing, 2009.

[1165] S.-H. Myaeng, D.-H. Jang, M.-S. Kim, and Z.-C. Zhoo. A flexible model for retrieval of SGML documents. In *21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia*, pages 138–145, 1998.

[1166] G. Myers. A fast bit-vector algorithm for approximate string matching based on dynamic progamming. *Journal of the ACM*, 46(3):395–415, 1999.

[1167] J. L. Myers and A. D. Well. *Research Design and Statistical Analysis.* Lawrence Erlbaum, 2003. Second Edition, 508 pages.

[1168] MySpace. http://www.myspace.com/, 2003.

[1169] M. Najork and J. L. Wiener. Breadth-first crawling yields high-quality pages. In *Proceedings of the Tenth Conference on World Wide Web*, pages 114–118, Hong Kong, May 2001. Elsevier Science.

[1170] R. Nallapati. Discriminative models for information retrieval. In M. Sanderson, K. Järvelin, J. Allan, and P. Bruza, editors, *SIGIR*, pages 64–71, Sheffield, UK, July 2004. ACM Press.

[1171] M. R. Naphade and J. R. Smith. On the detection of semantic concepts at TRECVID. In *MULTIMEDIA '04: Proceedings of the 12th Annual ACM International Conference on Multimedia*, pages 660–667, New York, NY, USA, 2004. ACM Press.

[1172] P. Nardiello, F. Sebastiani, and A. Sperduti. Discretizing continuous attributes in adaboost for text categorization. In *Proceedings of the 25th European Conference on Advances in Information Retrieval*, pages 320–334, 2003.

[1173] A. Nation. Visualizing websites using a hierarchical table of contents browser: Webtoc. In *Proceedings of the Third Conference on Human Factors and the Web*, Denver, CO, 1997.

[1174] National Library of Medicine (NLM). UMLS - Unified Medical Language System. http://www.nlm.nih.gov/research/umls/about_umls.html, September, 2006.

[1175] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[1176] G. Navarro. NR-grep: a fast and flexible pattern matching tool. *Software Practice and Experience (SPE)*, 31:1265–1312, 2001.

[1177] G. Navarro. Indexing text using the Ziv-Lempel trie. *Journal of Discrete Algorithms*, 2(1):87–114, 2004.

[1178] G. Navarro and R. Baeza-Yates. A language for queries on structure and contents of textual databases. In *18th Annual International ACM SIGIR Con-*

ference on Research and Development in Information Retrieval, Seattle, Washington, USA, pages 93–101, 1995.

[1179] G. Navarro and R. Baeza-Yates. Proximal nodes: A model to query document databases by content and structure. *ACM Transactions on Information Systems*, 15(4):400–435, 1997.

[1180] G. Navarro and R. Baeza-Yates. Very fast and simple approximate string matching. *Information Processing Letters*, 72:65–70, 1999.

[1181] G. Navarro and R. Baeza-Yates. A hybrid indexing method for approximate string matching. *Journal of Discrete Algorithms (JDA)*, 1(1):205–239, 2000.

[1182] G. Navarro, R. Baeza-Yates, E. Barbosa, N. Ziviani, and W. Cunto. Binary searching with non-uniform costs and its application to text retrieval. *Algorithmica*, 27:145–169, 2000.

[1183] G. Navarro, J. Kitajima, B. A. Ribeiro-Neto, and N. Ziviani. Distributed generation of suffix arrays. In A. Apostolico and J. Hein, editors, *Proc. of Combinatorial Pattern Matching*, number 1264 in LNCS, pages 102–115, Aarhus, Denmark, 1997. Springer-Verlag.

[1184] G. Navarro and V. Mäkinen. Compressed full-text indexes. *ACM Comput. Surv.*, 39(1), 2007.

[1185] G. Navarro, E. Moura, M. Neubert, N. Ziviani, and R. Baeza-Yates. Adding compression to block addressing inverted indexes. *Information Retrieval*, 3(1):49–77, 2000.

[1186] G. Navarro and M. Raffinot. Fast and flexible string matching by combining bit-parallelism and suffix automata. *ACM Journal of Experimental Algorithmics (JEA)*, 5(4), 2000.

[1187] G. Navarro and M. Raffinot. *Flexible Pattern Matching in Strings – Practical on-line search algorithms for texts and biological sequences.* Cambridge University Press, 2002. ISBN 0-521-81307-7. 280 pages.

[1188] G. Navarro and M. Raffinot. New techniques for regular expression searching. *Algorithmica*, 41(2):89–116, 2005.

[1189] G. Navarro and J. Tarhio. LZgrep: A Boyer-Moore string matching tool for Ziv-Lempel compressed text. *Software Practice and Experience (SPE)*, 35(12):1107–1130, 2005.

[1190] NDLTD. Networked Digital Library of Theses and Dissertations. http://www.ndltd.org, 2004.

[1191] M. Needleman. The shibboleth authentication/authorization system. *Serials Review*, 30(3):252–253, 2004.

[1192] M. Nelson. Data compression with the burrows-wheeler transform. *Dr. Dobb's Journal*, Sept. 1996.

[1193] M. L. Nelson and K. Maly. Buckets: smart objects for digital libraries. *Communications of the ACM*, 44(5):60–62, 2001.

[1194] M. L. Nelson, K. Maly, M. Zubair, and S. N. T. Shen. Soda: Smart objects, dumb archives. In *Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libraries (ECDL'99)*, pages 453–464, 1999.

[1195] J. Nesbit. The accuracy of approximate string matching algorithms. *J. of Computer-Based Instruction*, 13(3):80–83, 1986.

[1196] Netcraft. Web Server Survey. http://news.netcraft.com/, 2010.

[1197] M. L. Neufeld and M. Cornog. Database history: from dinosaurs to compact discs. *Journal of the American Society for Information Science*, 37(4):183–190, 1986.

[1198] T. Neumann, M. Bender, S. Michel, and G. Weikum. A reproducible benchmark for P2P retrieval. In *Proc. First Int. Workshop on Performance and Evaluation of Data Management Systems, ExpDB*, 2006.

[1199] M. E. J. Newman. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46:323-351, December 2005.

[1200] A. Y. Ng, A. X. Zheng, and M. I. Jordan. Link analysis, eigenvectors and stability. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 903-910, Seattle, Washington, USA, 2001.

[1201] H. Ng, W. Goh, and K. Low. Feature selection, perceptron learning, and a usability case study for text categorization. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 67-73, 1997.

[1202] D. Ngu and X. Wu. SiteHelper: a localized agent that helps incremental exploration of the World Wide Web. In *6th. Int'l. WWW Conference*, Santa Clara, CA, USA, Apr. 1997.

[1203] L. T. Nguyen, W. G. Yee, and O. Frieder. Adaptive distributed indexing for structured peer-to-peer networks. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 1241-1250, New York, NY, USA, 2008. ACM.

[1204] J. Nielsen. *Usability Engineering*. Academic Press, 1993.

[1205] J. Nielsen. Statistics for traffic referred by search engines and navigation directories to USEIT. http://www.useit.com/about/searchreferrals.html, 2004.

[1206] J. Nielsen. When Search Engines Become Answer Engines, 2004. http://www.useit.com/alertbox/20040816.html.

[1207] K. Nigam, A. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103-134, 2000.

[1208] G. Noether. Why Kendall Tau? Technical report, RSSCSE, 2008. http://rsscse.org.uk/ts/bts/noether/text.html.

[1209] G. Notess. Search Engines Showdown: The User's Guide to Search Engines. http://www.searchengineshowdown.com/, 1998.

[1210] H. Nottelmann and N. Fuhr. Evaluating different methods of estimating retrieval quality for resource selection. In *Proc. of the ACM Int'l Conf. on Information Retrieval*, pages 290-297, 2003.

[1211] H. Nottelmann and N. Fuhr. Combining CORI and the decision-theoretic approach for advanced resource selection. In *ECIR*, Sunderalnd, UK, 2004.

[1212] NTCIR—NII Test Collection for IR Project, 2009.

[1213] NTCIR-7 PATMT—Patent Translation Test Collection, 2009.

[1214] A. Ntoulas and J. Cho. Pruning Policies for Two-Tiered Inverted Index with Correctness Guarantee. In *SIGIR'07: Proceedings of the 30th International ACM SIGIR conference on Research and Development in Information Retrieval*, Amsterdam, The Netherlands, 2007.

[1215] A. Ntoulas, J. Cho, and C. Olston. What's new on the web?: the evolution of the Web from a search engine perspective. In *Proceedings of the 13th international conference on World Wide Web, WWW 2004, New York, NY, USA, May 17-20, 2004*, pages 1-12, 2004.

[1216] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam Web pages through content analysis. In *Proceedings of the World Wide Web conference*, pages 83-92, Edinburgh, Scotland, May 2006.

[1217] Nutch. http://lucene.apache.org/nutch/, 2007.

[1218] OCLC. Web Services and SRW/U, 2006. http://www.oclc.org/research/projects/webservices/default.htm.

[1219] V. L. O'Day and R. Jeffries. Orienteering in an information landscape: how information seekers get from here to there. In *Proceedings of the INTERCHI Conference on Human Factors in Computing Systems (CHI'93)*, Amsterdam, April 1993. IOS Press.

[1220] Open directory project: http://www.dmoz.org/, 2009.

[1221] C. of the ACM. Digital Libraries, April 1995. 38(4).

[1222] C. of the ACM. Digital Libraries: Global Scope, Unlimited Access, April 1998. 41(4).

[1223] C. of the ACM. Digital Libraries, May 2001. 44(5).

[1224] Y. Ogawa, T. Morita, and K. Kobayashi. A fuzzy document retrieval system using the keyword connection matrix and a learning method. *Fuzzy Sets and Systems*, 39:163–179, 1991.

[1225] P. Ogilvie and J. Callan. Combining document representations for known-item search. In *Proceedings of ACM SIGIR '03*, pages 143–150, New York, NY, USA, 2003. ACM Press.

[1226] P. Ogilvie and M. Lalmas. Investigating the exhaustivity dimension in content-oriented XML element retrieval evaluation. In *ACM CIKM International Conference on Information and Knowledge Management, Arlington, Virginia, USA*, pages 84–93, 2006.

[1227] R. A. O'Keefe and A. Trotman. The simplest query language that could possibly work. In *INEX 2003 Proceedings*, pages 167–174, 2003.

[1228] K. Olsen, R. Korfhage, K. Sochats, M. Spring, and J. Williams. Visualization of a Document Collection with Implicit and Explicit Links-The Vibe System. *Scandinavian Journal of Information Systems*, 5:79–95, 1993.

[1229] C. Olston and M. Najork. Web crawling. *Foundations and Trends in Information Retrieval*, 4(3):172–246, 2009.

[1230] E. O'Neill, B. Lavoie, and P. McClain. OCLC Web characterization project (position paper). In *Web Characterization Workshop*, Boston, USA, Nov 1998. http://www.w3.org/1998/11/05/WC-workshop/Papers/oneill.htm.

[1231] Open linking data project: http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData, 2007.

[1232] N. Orio. Music retrieval: A tutorial and review. *Foundations and Trends in Information Retrieval*, 1(1):1–90, 2006.

[1233] B. O'Riordan, K. Curran, and D. Woods. Investigating text input methods for mobile phones. *Journal of Computer Science*, 1(2):189–199, 2005.

[1234] Orkut. http://www.orkut.com/, 2004.

[1235] S. Orlando, R. Perego, and F. Silvestri. Design of a Parallel and Distributed WEB Search Engine. In *Proceedings of Parallel Computing (ParCo) 2001 conference*, pages 197–204. Imperial College Press, September 2001.

[1236] M. Özsu and L. Liu, editors. *Encyclopedia of Database Systems*. Springer, 2009.

[1237] T. Paek, S. Dumais, and R. Logan. WaveLens: A new view onto Internet search results. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'04)*, pages 727-734, 2004.

[1238] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: bringing order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.

[1239] G. Panagopoulos and C. Faloutsos. Bit-sliced signature files for very large text databases on a parallel machine architecture. In *Proc. 4th Inter. Conf. on Extending Database Technology (EDBT)*, number 779 in LNCS, pages 379–392, London, 1994. Springer-Verlag.

[1240] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2006.

[1241] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP-02, 7th Conference on Empirical Methods in Natural Language Processing*, pages 79–86, Philadelphia, US, 2002. Association for Computational Linguistics, Morristown, US.

[1242] G. Pant, K. Tsioutsiouliklis, J. Johnson, and C. L. Giles. Panorama: extending digital libraries with topical crawlers. In *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 142–150, Tucson, Arizona, 2004.

[1243] A. Papadopoulos and Y. Manolopoulos. Performance of nearest neighbor queries in R-trees. In F. N. Afrati and P. Kolaitis, editors, *Proc. of 6th Int. Conf. on Database Theory*, number 1186 in LNCS, pages 394–408, Delphi, Greece, Jan 1997.

[1244] J. Park and J. Kim. Effects of contextual navigation aids on browsing diverse web systems. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 257–264, New York, NY, USA, 2000. ACM.

[1245] N. Paskin. The DOI Handbook. Edition 4.2.0. International DOI Foundation (IDF). http://www.doi.org/hb.html, 1994.

[1246] A. Patterson. Why writing your own search engine is hard. *ACM Queue*, April 2004.

[1247] E. Patterson, E. Roth, and D. Woods. Predicting Vulnerabilities in Computer-Supported Inferential Analysis under Data Overload. *Cognition, Technology & Work*, 3(4):224 237, 2001.

[1248] V. Paxson. End-to-end routing behavior in the Internet. *ACM SICOMM Computer Communication Review*, 35(5):43-56, October 2006.

[1249] S. Payette and T. Staples. The Mellon Fedora Project. In *Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries (ECDL'02)*, pages 406 421, Rome, Italy, 2002.

[1250] G. W. Paynter. Developing practical automatic metadata assignment and evaluation tools for Internet resources. In M. Marlino, T. Sumner, and F. M. S. III, editors, *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2005, Denver, CA, USA, June 7-11, 2005, Proceedings*, pages 291–300. ACM, 2005.

[1251] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, Inc., 1988.

[1252] J. Pehcevski and B. Piwowarski. Evaluation metrics for structured text retrieval. In *Encyclopedia of Database Systems*. Springer, 2009.

[1253] J. Pehcevski and J. A. Thom. Hixeval: Highlighting XML retrieval evaluation. In *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005*, pages 43–57, Dagstuhl Castle, Germany, 2006. Revised Selected Papers.

[1254] D. A. Pereira, B. A. Ribeiro-Neto, N. Ziviani, and A. H. F. Laender. Using Web information for creating publication venue authority files. In *JCDL*, pages 295–304, 2008.

[1255] D. A. Pereira, B. A. Ribeiro-Neto, N. Ziviani, A. H. F. Laender, M. A. Gonçalves, and A. A. Ferreira. Using Web information for author name disambiguation. In *JCDL*, pages 49–58, 2009.

[1256] A. Perkins. The classification of search engine spam. Available online at http://www.silverdisc.co.uk/articles/spam-classification/, September 2001.

[1257] M. Persin. Document filtering for fast ranking. In *Proc. of the 17th ACM SIGIR Conference*. Springer Verlag, 1994.

[1258] M. Persin, J. Zobel, and R. Sacks-Davis. Filtered document retrieval with frequency-sorted indexes. *Journal of the American Society for Information Science*, 47(10):749–764, Oct. 1996.

[1259] S. Perugini, M. A. Gonçalves, and E. A. Fox. Recommender systems research: A connection-centric survey. *Journal of Intelligent Information Systems*, 23(2):107–143, 2004.

[1260] S. Perugini, K. McDevitt, R. Richardson, M. Perez-Quinones, R. Shen, N. Ramakrishnan, C. Williams, and E. A. Fox. Enhancing Usability in CITIDEL: Multimodal, Multilingual, and Interactive Visualization Interfaces. In *Proc. of the 4th Joint Conf. on Digital Libraries (JCDL'2004)*, pages 315–324, Tucson, Arizona, June 7-11, 2004.

[1261] N. Pharo and A. Trotman. The use case track at INEX 2006. *SIGIR Forum*, 41(1):64–66, 2007.

[1262] D. Pierrakos, G. Paliouras, C. Papatheodorou, and C. D. Spyropoulos. Web usage mining as a tool for personalization: A survey. *User Modeling and User-Adapted Interaction*, 13(4):311–372, 2003.

[1263] D. Pimienta. Languages, culture, and Internet (in French). http://funredes.org/, March 1998.

[1264] K. Pinel-Sauvagnat. Propagation-based structured text retrieval. In *Encyclopedia of Database Systems*. Springer, 2009.

[1265] B. Pinkerton. `comp.infosystems.announce` newsgroup, June 1994.

[1266] B. Pinkerton. Finding what people want: Experiences with the WebCrawler. In *Proceedings of the first World Wide Web Conference*, Geneva, Switzerland, May 1994.

[1267] S. Piontek and K. Garlock. Creating a World Wide Web resource collection. *Internet Research: Electronic Networking Applications and Policy*, 6(4):20–26, 1996.

[1268] P. Pirolli. Computational models of information scent-following in a very large browsable text collection. In *CHI*, pages 3–10, 1997.

[1269] P. Pirolli. *Information Foraging Theory*. Oxford University Press, 2007.

[1270] P. Pirolli and S. Card. Information foraging models of browsers for very large document spaces. In *Advanced Visual Interfaces*, L'Aquila, Italy, May 1998.

[1271] P. Pirolli and S. Card. Information foraging. *Psychological Review*, 106(4):643–675, 1999.

[1272] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of the 2005 International Conference on Intelligence Analysis*, McClean, VA, May 2005.

[1273] P. Pirolli, J. Pitkow, and R. Rao. Silk from a sow's ear: Extracting usable structures from the Web. In *Proc. of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 118–125, Zurich, Switzerland, May 1996. ACM Press.

[1274] J. Pitkow and K. Bharat. WebViz: A tools for World Wide Web access log analysis. In *Proc. of the First International World Wide Web Conference*, Geneva, Switzerland, May 1994. http://www1.cern.ch/PapersWWW94/pitkow-webvis.ps.

[1275] J. Pitkow, H. Schütze, T. Cass, R. Cooley, D. Turnbull, A. Edmonds, E. Adar, and T. Breuel. Personalized search. *Commun. ACM*, 45(9):50–55, 2002.

[1276] B. Piwowarski and G. Dupret. Evaluation in (XML) information retrieval: expected precision-recall with user modelling (EPRUM). In *29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA*, pages 260–267, 2006.

[1277] B. Piwowarski, G. Dupret, and R. Jones. Mining user Web search activity with layered Bayesian networks or how to capture a click in its context. In R. Baeza-Yates, P. Boldi, B. A. Ribeiro-Neto, and B. B. Cambazoglu, editors, *WSDM*, pages 162–171, Barcelona, Spain, February 2009. ACM.

[1278] B. Piwowarski and M. Lalmas. Providing consistent and exhaustive relevance assessments for XML retrieval evaluation. In *12th ACM international conference on Information and knowledge management, Washington, DC, USA*, pages 361–370, 2004.

[1279] B. Piwowarski, A. Trotman, and M. Lalmas. Sound and complete relevance assessments for XML retrieval. *ACM Transactions in Information Systems*, 27(1), 2008.

[1280] B. Piwowarski and H. Zaragoza. Predictive user click models based on click-through history. In M. J. Silva, A. H. F. Laender, R. Baeza-Yates, D. L. McGuinness, B. Olstad, Ø. H. Olsen, and A. O. Falcão, editors, *CIKM*, pages 175–182, Lisbon, Portugal, November 2007. ACM.

[1281] C. Plaisant, B. Shneiderman, K. Doan, and T. Bruns. Interface and data architecture for query preview in networked information systems. *ACM Transactions on Information Systems (TOIS)*, 17(3):320–341, 1999.

[1282] B. Poblete and R. Baeza-Yates. Query-sets: using implicit feedback and query patterns to organize Web documents. In J. Huai, R. Chen, H.-W. Hon, Y. Liu, W.-Y. Ma, A. Tomkins, and X. Zhang, editors, *Proceedings of the 17th International Conference on World Wide Web, WWW 2008*, pages 41–50, Beijing, China, April 2008. ACM Press.

[1283] B. Poblete, M. Spiliopoulou, and R. Baeza-Yates. Website privacy preservation for query log publishing. In *Proceedings of the First SIGKDD International Workshop on Privacy, Security, and Trust in KDD (PinKDD'07), Lecture Notes in Computer Science*, volume 4890. Springer, 2008.

[1284] S. Podlipnig and L. Boszormenyi. A survey of Web cache replacement strategies. *ACM Computing Surveys*, 35(4):374–398, 2003.

[1285] I. Podnar, M. Rajman, T. Luu, F. Klemm, and K. Aberer. Scalable Peer-to-Peer Web Retrieval with Highly Discriminative Keys. In *ICDE'07: Proceedings of the 23nd International conference on Data Engineering*, Istambul, Turkey, 2007.

[1286] P.Ogilvie and J. Callan. Hierarchical language models for XML component retrieval. In *Advances in XML Information Retrieval, Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*, pages 224–237, Dagstuhl Castle, Germany, 2005. Revised Selected Papers.

[1287] C. A. Pogue and P. Willet. Use of text signatures for document retrieval in a highly parallel environment. *Parallel Computing*, 4:259–268, 1987.

[1288] A. Pollock and A. Hockley. What's Wrong with Internet Searching. *D-Lib Magazine*, 1997. http://www.dlib.org.

[1289] D. B. Ponceleón and A. Dieberger. Hierarchical brushing in a collection of video data. In *Proceedings of Hawaii International Conference on System Science (HICSS)*, Maui, HI, 2001.

[1290] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, 1998.

[1291] E. Popovici, G. Ménier, and P.-F. Marteau. SIRIUS XML IR System at INEX 2006: Approximate Matching of Structure and Textual Content. In *Comparative Evaluation of XML Information Retrieval Systems, 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006*, pages 185–199, Dagstuhl Castle, Germany, 2007. Revised and Selected Papers.

[1292] M. Porter. An algorithm for suffix striping. In K. S. Jones and P. Willet, editors, *Readings in Information Retrieval*, pages 313–316. Morgan Kaufmann Publishers, Inc., 1997.

[1293] B. Pôssas, N. Ziviani, and W. Meira. Enhancing the set-based model using proximity information. In *SPIRE - 9th Int. Symposium on String Processing and Information Retrieval*, pages 104–116, 2002. Lisbon, Portugal.

[1294] B. Pôssas, N. Ziviani, W. Meira, and B. A. Ribeiro-Neto. Set-based model: A new approach to information retrieval. In *25th ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237, 2002. Tampere, Finland.

[1295] B. Pôssas, N. Ziviani, W. Meira, and B. A. Ribeiro-Neto. Set-based vector model: An efficient approach for correlation-based ranking. *ACM Transactions on Information Systems*, 23(4):397–429, 2005.

[1296] B. Pôssas, N. Ziviani, B. A. Ribeiro-Neto, and W. Meira. Processing conjunctive and phrase queries with the set-based model. In *SPIRE - 11th Int. Symposium on String Processing and Information Retrieval*, pages 171–183, 2004. Padova, Italy.

[1297] B. Pôssas, N. Ziviani, B. A. Ribeiro-Neto, and W. Meira Jr. Maximal termsets as a query structuring mechanism. In *CIKM*, pages 287–288, 2005.

[1298] G. Potamianos, C. Neti, G. Gravier, A. Garg, and A. W. Senior. Automatic recognition of audio-visual speech: Recent progress and challenges. *Proceedings of the IEEE*, 91(9), September 2003.

[1299] A. L. Powell and J. C. French. Growth and server availability of the NCSTRL digital library. In *Proceedings of the 5th ACM International Conference on Digital Libraries*, pages 264–265, San Antonio, Texas, 2000.

[1300] A. L. Powell and J. C. French. Comparing the performance of collection selection algorithms. *ACM Trans. Inf. Syst.*, 21(4):412–456, 2003.

[1301] W. Pratt, M. Hearst, and L. Fagan. A knowledge-based approach to organizing retrieved documents. In *Proceedings of 16th Annual Conference on Artificial Intelligence(AAAI 99)*, Orlando, FL, 1999.

[1302] A. Pretschner and S. Gauch. Ontology based personalized search. In *ICTAI '99: Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*, page 391, Washington, DC, USA, 1999. IEEE Computer Society.

[1303] P. Proulx, S. Tandon, A. Bodnar, D. Schroh, W. Wright, D. Schroh, R. Harper, and W. Wright. Avian Flu Case Study with nSpace and GeoTime. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST'06)*. IEEE, 2006.

[1304] D. Puppin, F. Silvestri, and D. Laforenza. Query-driven document partitioning and collection selection. In *INFOSCALE 2006: Proceedings of the first International Conference on Scalable Information Systems*, 2006.

[1305] The PURL Team, Persistent Uniform Resource Locator (PURL). http://purl.oclc.org/.

[1306] J. Qin, Y. Zhou, and M. Chau. Building domain-specific Web collections for scientific digital libraries: a meta-search enhanced focused crawling method. In *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2004, Proceedings*, pages 135–141, Tuscon, AZ, USA, 2004.

[1307] T. Qin, X. D. Zhang, M. F. Tsai, D. S. Wang, T. Y. Liu, and H. Li. Query-level loss functions for information retrieval. *Information Processing & Management*, 44(2):838-855, 2008.

[1308] T. Qin, X.-D. Zhang, D.-S. Wang, T.-Y. Liu, W. Lai, and H. Li. Ranking with multiple hyperplanes. In *SIGIR 2007: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, July 23-27, 2007*, pages 279–286, 2007.

[1309] Y. Qiu and H. Frei. Concept based query expansion. In *Proc. ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 160–169, Pittsburgh, PA, USA, 1993.

[1310] Qos project. http://qos.sourceforge.net/, 2007.

[1311] J. Quinlan. Discovering rules by induction from large collections of examples. In *Expert Systems in the Micro Electronic Age*, Edinburgh, UK, 1979. Edinburgh University Press.

[1312] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[1313] J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers Inc., San Mateo, CA, 1993.

[1314] T. Radecki. Mathematical model of information retrieval system based on the concept of fuzzy thesaurus. *Information Processing & Management*, 12:313–318, 1976.

[1315] T. Radecki. Mathematical model of time-effective information retrieval system based on the theory of fuzzy sets. *Information Processing & Management*, 13:109–116, 1977.

[1316] T. Radecki. Fuzzy set theoretical approach to document retrieval. *Information Processing & Management*, 15:247–259, 1979.

[1317] T. Radecki. On the inclusiveness of information retrieval systems with documents indexed by weighted descriptors. *Fuzzy Sets and Systems*, 5:159–176, 1981.

[1318] T. Radecki. Trends in research on information retrieval—the potential for improvements in conventional Boolean retrieval systems. *Information Processing & Management*, 24:219–227, 1988.

[1319] F. Radlinski, A. Broder, P. Ciccolo, E. Gabrilovich, V. Josifovski, and L. Riedel. Optimizing relevance and revenue in ad search: a query substitution approach. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 403–410, New York, NY, USA, 2008. ACM.

[1320] F. Radlinski and T. Joachims. Query chains: learning to rank from implicit feedback. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 239-248, 2005.

[1321] F. Radlinski and T. Joachims. Minimally invasive randomization for collecting unbiased preferences from clickthrough logs. In *Conference of the Association for the Advancement of Artificial Intelligence (AAAI)*, pages 1406–1412, 2006.

[1322] F. Radlinski and T. Joachims. Active exploration for learning rankings from clickthrough data. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2007.

[1323] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *Conference on Information and Knowledge Management (CIKM)*, 2008.

[1324] S. Raghavan and H. Garcia-Molina. Crawling the hidden web. In *Proceed-*

ings of the Twenty-seventh International Conference on Very Large Databases (VLDB), pages 129–138, Rome, Italy, 2001. Morgan Kaufmann.

[1325] V. Raghavan, P. Bollmann, and G. Jung. Retrieval system evaluation using recall and precision: Problems and answers. In *Proceedings of the 12th ACM SIGIR Conference*, pages 59–68, 1989.

[1326] V. Raghavan, G. Jung, and P. Bollmann. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transactions on Office and Information Systems*, 7(3):205–229, 1989.

[1327] V. Raghavan and S. Wong. A critical analysis of vector space model for information retrieval. *Journal of the American Society for Information Sciences*, 37(5):279 287, 1986.

[1328] V. V. Raghavan, P. Bollmann, and G. S. Jung. Retrieval system evaluation using recall and precision: Problems and answers. In *12th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Cambridge, Massachusetts, USA*, pages 59 68, 1989.

[1329] V. V. Raghavan and H. Sever. On the reuse of past optimal queries. *18th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 344–350, 1995.

[1330] C. Raiciu, F. Huici, M. Handley, and D. S. Rosenblum. Roar: increasing the flexibility and performance of distributed search. *SIGCOMM Comput. Commun. Rev.*, 39(4):291–302, 2009.

[1331] G. Ramírez. *Structural Features in XML Retrieval*. PhD thesis, University of Amsterdam, 2007.

[1332] G. Ramírez. Processing overlaps. In *Encyclopedia of Database Systems*. Springer, 2009.

[1333] K. H. Randall, R. Stata, J. L. Wiener, and R. G. Wickremesinghe. The link database: Fast access to graphs of the web. In *DCC '02: Proceedings of the Data Compression Conference (DCC '02)*, page 122, Washington, DC, USA, 2002. IEEE Computer Society.

[1334] E. Rasmussen. Clustering algorithms. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Data Structures & Algorithms*, pages 419–442. Prentice Hall, Englewood Cliffs, NJ, USA, 1992.

[1335] Y. Rasolofo, D. Hawking, and J. Savoy. Result merging strategies for a current news metasearcher. *Information Processing and Management*, 39:581–609, 2003. http://david-hawking.net/pubs/rasolofo_ipm03.pdf.

[1336] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 161–172, New York, NY, USA, 2001. ACM.

[1337] A. Rauber, G. Widmer, S. Downie, S. Dixon, and D. Bainbridge, editors. *Proceedings International Symposium for Audio Information Retrieval (ISMIR)*. Austrian Computer Society, Vienna, Austria, October 2007.

[1338] J. Ray, R. Dale, R. Moore, V. Reich, W. Underwood, McCray, and A. T. Panel on digital preservation. In *JCDL'02: Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 365–367, 2002.

[1339] K. Rayner. Eye movements in reading and information processing. *Psychological Bulletin*, 124:372–252, 1998.

[1340] R. Reddy and I. Wladawsky-Berger. Digital Libraries: Universal Access to Human Knowledge - A Report to the President. President's Information Technology Advisory Committee (PITAC), Panel on Digital Libraries. http://www.itrd.gov/pubs/pitac/pitacdl-9feb01.pdf, 2001.

[1341] W. J. Reed. The Pareto, Zipf and Other Power Laws. *Economics Letters*, 74(15-19), 2001.

[1342] H. Reiterer, G. Tullius, and T. Mann. Insyder: a content-based visual-information-seeking system for the web. *International Journal on Digital Libraries*, pages 25–41, Mar 2005.

[1343] P. Resnick and H. R. Varian. Recommender systems. *Commun. ACM*, 40(3):56–58, 1997.

[1344] M. Rettig. Prototyping for Tiny Fingers. *Communications of the Acm*, 37(4), 1994.

[1345] Reuters Corpus Volume 1 (RCV1), 2000. Produced by Reuters Ltd., RCV1 is made available for use in research and development of natural language-processing, information-retrieval or machine learning systems.

[1346] Reuters Corpus Volume 2 (RCV2), 2005. Produced by Reuters Ltd., RCV2 is made available for use in research and development of natural language-processing, information-retrieval or machine learning systems.

[1347] D. Reynolds, T. Quatieri, and R. Dunn. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10(1-3):19–41, 2000.

[1348] P. Reynolds and A. Vahdat. Efficient Peer-to-Peer Keyword Searching. In *Middleware'03: Proceedings of the 4th International Middleware conference*, Rio de Janeiro, Brazil, 2003.

[1349] B. Ribeiro-Neto and R. R. Barbosa. Query performance for tightly coupled distributed digital libraries. In *Proc. 3rd ACM Conference on Digital Libraries*, pages 182–190, Pittsburgh, PA, June 1998. ACM Press, New York.

[1350] B. A. Ribeiro-Neto, M. Cristo, P. B. Golgher, and E. S. de Moura. Impedance coupling in content-targeted advertising. In *SIGIR*, pages 496–503, 2005.

[1351] B. A. Ribeiro-Neto, A. H. Laender, and L. R. de Lima. An experimental study in automatically categorizing medical documents. *Journal of the American Society for Information Science and Technology*, 52(5):391–401, 2001.

[1352] B. A. Ribeiro-Neto and R. Muntz. A belief network model for IR. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Modeling, pages 253-260, 1996.

[1353] B. A. Ribeiro-Neto and R. R. Muntz. Fuzzy ranking of approximate answers. In *Second Int. Conference on Flexible Query Answering Systems (FQAS)*, pages 41–56, 1996.

[1354] I. E. G. Richardson. *H.264 and MPEG-4 Video Compression: Video Coding for Next-Generation Multimedia*. Wiley and Sons, 2003.

[1355] S. Rieh. Judgment of information quality and cognitive authority in the Web. *Journal of the American Society for Information Science and Technology*, 53(2):145–161, 2002.

[1356] S. Y. Rieh and H. I. Xie. Analysis of multiple query reformulations on the web: The interactive information retrieval context. *Information Processing & Management*, 42(3):751–768, 2006.

[1357] J. Rissanen and G. G. Langdon. Arithmetic coding. *IBM Journal of Research and Development*, 23:149–162, 1979.

[1358] J. Risson and T. Moors. Survey of research towards robust peer-to-peer networks: search methods. *Comput. Netw.*, 50(17):3485–3521, 2006.

[1359] K. M. Risvik. *Scaling Internet Search Engines: Methods and Analysis*. PhD thesis, Norwegian University of Science and Technology, 2004.

[1360] K. M. Risvik, Y. Aasheim, and M. Lidal. Multi-tier architecture for Web search engines. In *LA-WEB*, pages 132 143, Santiago, Chile, 2003.

[1361] K. M. Risvik and R. Michelsen. Search engines and Web dynamics. *Computer Networks*, 39(3):289–302, June 2002.

[1362] RLG/NARA audit checklist for certifying a trusted digital repository, 2005. http://www.rlg.org/en/pdfs/rlgnara-repositorieschecklist.pdf.

[1363] P. A. Roberto, R. L. T. Santos, M. A. Gonçalves, and A. H. F. Laender. On RDBMS and workflow support for componentized digital libraries. In *XXI Simpósio Brasileiro de Banco de Dados*, pages 87 101, Florianópolis, Santa Catarina, Brasil, October 2006.

[1364] S. Robertson. The probability ranking principle in IR. *Journal of Documentation*, pages 294–304, 1977.

[1365] S. Robertson and K. S. Jones. Relevance weighting of search terms. *Journal of the American Society for Information Sciences*, 27(3):129–146, 1976.

[1366] S. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau. Okapi at TREC. In D. K. Harman, editor, *The First Text REtrieval Conference (TREC-1)*, pages 21–30, Gaithersburg, MD, USA, 1993. Dept. of Commerce, National Institute of Standards and Technology.

[1367] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-2. In D. K. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, pages 21–34, Gaithersburg, MD, USA, 1994. Dept. of Commerce, National Institute of Standards and Technology.

[1368] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, pages 109–128, Gaithersburg, MD, USA, 1995. Dept. of Commerce, National Institute of Standards and Technology.

[1369] S. Robertson and H. Zaragoza. The probabilistic relevance model: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

[1370] S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 42–49, New York, NY, USA, 2004. ACM.

[1371] S. E. Robertson. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of Documentation*, 60(5):503–520, 2004.

[1372] S. E. Robertson and M. M. Hancock-Beaulieu. On the evaluation of IR systems. *Inf. Process. Manage.*, 28(4):457–466, 1992.

[1373] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR*, pages 232–241, 1994.

[1374] S. E. Robertson and S. Walker. On relevance weights with little relevance information. In *ACM SIGIR '97: Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 16–24, New York, NY, USA, 1997.

[1375] J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice-Hall Inc, Englewood Cliffs, New Jersey, 1971.

[1376] K. Rodden, W. Basalaj, D. Sinclair, and K. R. Wood. Does organisation by similarity assist image browsing? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'01)*, pages 190–197, 2001.

[1377] P. Roget. *Roget's II the New Thesaurus*. Houghton Mifflin Company, Boston, USA, 1988.

[1378] H. L. Roitblat. Information retrieval and eDiscovery, 2006. http://www.ediscoveryinstitute.org/pubs/InformationRetrievalandeDiscovery.pdf.

[1379] T. Rölleke, M. Lalmas, G. Kazai, I. Ruthven, and S. Quicker. The accessibility dimension for structured document retrieval. In *Advances in Information Retrieval, 24th BCS-IRSG European Colloquium on IR Research, Glasgow, UK*, pages 284–302, 2002.

[1380] D. Rose, D. Orr, and R. Kantamneni. Summary attributes and perceived search quality. In *Proceedings of the 16th International Conference on World Wide Web (WWW'07)*, pages 1201–1202. ACM Press New York, NY, USA, 2007.

[1381] D. E. Rose and R. K. Belew. Legal information retrieval a hybrid approach. In *ICAIL '89: Proceedings of the 2nd international conference on Artificial intelligence and law*, pages 138–146, New York, NY, USA, 1989. ACM.

[1382] D. E. Rose and D. Levinson. Understanding user goals in Web search. In *Proc. of the 14th international conference on World Wide Web*, pages 13–19. ACM Press, 2004.

[1383] L. Rosenfeld and M. Hurst. *Search Analytics: Conversations with your customers*. Rosenfeld Media, 2010. http://www.rosenfeldmedia.com/books/searchanalytics/.

[1384] R. Rosenfeld. Two decades of statistical language modeling: where do we go from here? *Proceedings of the IEEE*, 88(8), 2000.

[1385] S. Ross. *Introduction to probability models*. Harcourt Academic Press, 2000.

[1386] S. Ross and M. Hedstrom. Preservation research and sustainable digital libraries. *Int. J. on Digital Libraries*, 5(4):317–324, 2005.

[1387] J. Rothenberg. *Using Emulation to Preserve Digital Documents*. Koninklijke Bibliotheek, The Netherlands, Aug. 21 2000.

[1388] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *Proc. of ACM-SIGMOD*, pages 71–79, San Jose, CA, May 1995.

[1389] T. Rowlands, D. Hawking, and R. Sankaranarayana. Workload sampling for enterprise search evaluation. In *Proceedings of ACM SIGIR 2007*, pages 887–888, July 2007. Poster paper. http://es.csiro.au/pubs/rowlandsHS07.pdf.

[1390] J. Rowley. The controlled versus natural indexing languages debate revisited: a perspective on information retrieval practice and research. *Journal of Information Science*, 20(2):108–119, 1994.

[1391] A. I. T. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware '01: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, pages 329–350, London, UK, 2001. Springer-Verlag.

[1392] S. Rüger. *Multimedia Information Retrieval*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan Claypool, 2009.

[1393] Y. Rui and T. Huang. A novel relevance feedback technique in image retrieval. In *Proceedings of the seventh ACM international conference on Multimedia (Part 2)*, pages 67–70, 1999.

[1394] Y. Rui, T. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in mars. In *Proceedings. of the IEEE International Conference on Image Processing*, volume 2, pages 815–818, 1997.

[1395] Y. Rui, T. Huang, and S. Mehrotra. Relevance feedback techniques in interactive content-based image retrieval. In *Storage and Retrieval for Image and Video Databases (SPIE)*, 1998.

[1396] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Transactions*

on Circuits and Systems for Video Technology, 8(5):644–655, 1998.

[1397] M. Ruiz and P. Srinivazan. Hierarchical neural networks for text categorization. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 281–282, 1999.

[1398] RuleQuest Research. C5.0 data mining tool, 2008. http://www.rulequest.com/.

[1399] S. Russel and P. Norvig. *Artificial Intelligence - A Modern Approach.* Prentice-Hall, 2002.

[1400] D. Russell, M. Slaney, Y. Qu, and M. Houston. Being literate with large document collections: Observational studies and cost structure tradeoffs. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, 2006.

[1401] D. Russell, M. Stefik, P. Pirolli, and S. Card. The cost structure of sensemaking. In *Proceedings of the INTERCHI Conference on Human Factors in Computing Systems (CHI'93)*, Conceptual Analysis of Users and Activity, pages 269–276, 1993.

[1402] I. Ruthven and M. Lalmas. A Survey on the Use of Relevance Feedback for Information Access Systems. *The Knowledge Engineering Review*, 18(02):95–145, 2003.

[1403] W. Sachs. An approach to associative retrieval through the theory of fuzzy sets. *Journal of the American Society for Information Sciences*, pages 85–87, 1976.

[1404] K. Sadakane. Compressed text databases with efficient query algorithms based on the compressed suffix array. In *Proc. 11th International Symposium on Algorithms and Computation (ISAAC)*, LNCS v. 1969, pages 410–421, 2000.

[1405] K. Sadakane. New text indexing functionalities of the compressed suffix arrays. *Journal of Algorithms*, 48(2):294–313, 2003.

[1406] M. Sahami and T. D. Heilman. A web-based kernel function for measuring the similarity of short text snippets. *World Wide Web Conference*, pages 377–386, 2006.

[1407] Y. Saito and M. Shapiro. Optimistic replication. *ACM Computing Surveys*, 37(1):42–81, March 2005.

[1408] G. Salton. *The SMART Retrieval System - Experiments in Automatic Document Processing.* Prentice-Hall Inc, Englewood Cliffs, New Jersey, 1971.

[1409] G. Salton and C. Buckley. Parallel text search methods. *Commun. ACM*, 31(2):202–215, Feb. 1988.

[1410] G. Salton and C. Buckley. Term-weighting approaches in automatic retrieval. *Information Processing & Management*, 24(5):513–523, 1988.

[1411] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–297, 1990.

[1412] G. Salton, E. A. Fox, and H. Wu. Extended Boolean information retrieval. *Communications of the ACM*, 26(11):1022–1036, Nov. 1983.

[1413] G. Salton and M. E. Lesk. Computer evaluation of indexing and text processing. *Journal of the ACM*, 15(1):8–36, Jan. 1968.

[1414] G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Co., New York, 1983.

[1415] G. Salton, A. Singhal, C. Buckley, and M. Mitra. Automatic text decomposition using text segments and text themes. In *Proceedings of the the seventh ACM conference on Hypertext*, pages 53–65, 1996.

[1416] G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[1417] G. Salton, C. Yang, and C. Yu. A theory of term importance in automatic text analysis. *Journal of the American Society for Information Sciences*, 26(1):33–44, 1975.

[1418] G. Salton and C. S. Yang. On the specification of term values in automatic indexing. *Journal of Documentation*, 29:351–372, 1973.

[1419] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Computer Graphics and Geometric Modeling. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[1420] P. Sanders and F. Transier. Intersection in integer inverted indices. In *ALENEX'07*, pages 71–83, 2007.

[1421] T. Sanders. Personal communication, 1993.

[1422] M. Sanderson. Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval*, 4(4):247–375, 2010.

[1423] R. L. T. Santos, P. A. Roberto, M. A. Gonçalves, and A. H. F. Laender. Design, implementation, and evaluation of a wizard tool for setting up component-based digital libraries. In *Research and Advanced Technology for Digital Libraries, 10th European Conference, ECDL 2006, Alicante, Spain, September 17-22, 2006, Proceedings*, volume 4172, pages 135–146, 2006.

[1424] T. Saracevic. Evaluation of evaluation in information retrieval. In *SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 138–146, 1995.

[1425] T. Saracevic. Digital library evaluation: Toward evolution of concepts. *Library Trends*, 49(2):350–369, 2000.

[1426] K. Sauvagnat, M. Boughanem, and C. Chrisment. Answering content and structure-based queries on XML documents using relevance propagation. *Information Systems*, 31(7):621–635, 2006.

[1427] K. Sauvagnat, L. Hlaoua, and M. Boughanem. XFIRM at INEX 2005: Ad-Hoc and Relevance Feedback Tracks. In *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005*, pages 88–103, Dagstuhl Castle, Germany, 2006. Revised Selected Papers.

[1428] H. Sawhney, R. Kumar, G. Gendel, J. Bergen, D. Dixon, and V. Paragano. VideoBrushTM: experiences with consumer video mosaicing. In *Fourth IEEE Workshop on Applications of Computer Vision, 1998. WACV '98*, pages 56–62, Oct 1998.

[1429] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

[1430] R. E. Schapire. The boosting approach to machine learning: An overview, December 2002. www.cs.princeton.edu/~schapire/boost.html.

[1431] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[1432] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[1433] R. E. Schapire, Y. Singer, and A. Singhal. Boosting and Rocchio applied to text filtering. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 215–223, Melbourne, Australia, 1998.

[1434] B. R. Schatz. Information Retrieval in Digital Libraries: Bringing Search to the Net. *Science*, 275:327–335, January 1997.

[1435] E. Scheirer and M. Slaney. Construction and evaluation of a robust multi-feature speech/music discriminator. In *ICASSP '97: Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97)-Volume 2*, page 1331, Washington, DC, USA, 1997. IEEE Computer Society.

[1436] R. Schenkel and M. Theobald. Structural Feedback for Keyword-Based XML Retrieval. In *Advances in Information Retrieval, 28th European Conference on IR Research, ECIR 2006, London, UK*, pages 326–337, 2006.

[1437] R. Schenkel and M. Theobald. Integrated DB&IR. In *Encyclopedia of Database Systems*. Springer, 2009.

[1438] T. Schlieder and H. Meuss. Querying and ranking XML documents. *JASIST*, 53(6):489–503, 2002.

[1439] B. Schlkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[1440] F. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys*, 22(4):299–319, December 1990.

[1441] K. Schneider. How OPACs suck, part 1: Relevance rank (or the lack of it), 2006. Available at http://www.techsource.ala.org/blog/2006/03/how-opacs-suck-part-1-relevance-rank-or-the-lack-of-it.html.

[1442] U. Schonfeld, Z. Bar-Yossef, and I. Keidar. Do not crawl in the DUST: different URLs with similar text. In *WWW'06: Proceedings of the 15th international conference on World Wide Web*, pages 1015–1016, New York, NY, USA, 2006. ACM Press.

[1443] M. Schroeder. *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*. W.H. Freeman and Company, New York, 1991.

[1444] H. Schutze, D. Hull, and J. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proc. of the 18th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 229–237, Seattle, WA, 1995.

[1445] E. S. Schwartz and B. Kallick. Generating a canonical prefix encoding. *Communications of the ACM*, 7:166–169, 1964.

[1446] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[1447] F. Sebastiani, A. Sperduti, and N. Valdambrini. An improved boosting algorithm and its application to text categorization. In *CIKM '00: Proceedings of the ninth international conference on Information and knowledge management*, pages 78–85, New York, NY, USA, 2000. ACM.

[1448] E. Selberg and O. Etzioni. Multi-service search and comparison using the MetaCrawler. In *Proc. of the Fourth International World Wide Web Conference*, Boston, Dec. 1995. http://www.w3.org/pub/Conferences/WWW4/Papers/169.

[1449] P. Sellers. The theory and computation of evolutionary distances: pattern recognition. *Journal of Algorithms*, 1:359–373, 1980.

[1450] P. Serdyukov, R. Aly, and D. Hiemstra. University of twente at the trec 2008 enterprise track: Using the global web as an expertise evidence source. In *Proceedings of TREC-2008*, 2009. http://trec.nist.gov/pubs/trec17/papers/utwente.ent.rev.pdf.

[1451] M. Á. Serrano, A. G. Maguitman, M. Boguñá, S. Fortunato, and A. Vespignani. Decoding the structure of the www: facts versus sampling biases, 2005.

[1452] C. E. Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:398–403, 1948.

[1453] J. Shapiro, V. G. Voiskunskii, and V. J. Frants. *Automated Information Retrieval : Theory and Text-Only Methods*. Academic Press, 1997.

[1454] W. Shaw, J. Wood, R. Wood, and H. Tibbo. The cystic fibrosis database: Content and research opportunities. *Library and Informatin Science Research*, 13:347-366, 1991.

[1455] W. Shaw Jr., R. Burgin, and P. Howell. Performance standards and evaluations in IR test collections: Cluster-based retrieval models. *Information Processing & Management*, 33(1):1–14, 1997.

[1456] W. Shaw Jr., R. Burgin, and P. Howell. Performance standards and evaluations in IR test collections: Vector-space and other retrieval models. *Information Processing & Management*, 33(1):15–36, 1997.

[1457] D. Shen, R. Pan, J.-T. Sun, J. J. Pan, K. Wu, J. Yin, and Q. Yang. Q^2C@UST: our winning solution to query classification in KDDCUP 2005. *SIGKDD Explorations*, 7(2):100–110, 2005.

[1458] R. Shen. *Applying the 5S Framework to Integrating Digital Libraries*. PhD thesis, Virginia Tech, 2005.

[1459] R. Shen, M. A. Gonçalves, W. Fan, and E. A. Fox. Requirements gathering and modeling of domain-specific digital libraries with the 5S framework: An archaeological case study with ETANA. In *Proceedings of the 9th European Conference on Research and Advanced Technology for Digital Libraries*, volume 3652, pages 1–12, Vienna, Austria, 2005.

[1460] R. Shen, N. S. Vemuri, W. Fan, and E. A. Fox. What is a successful digital library? In *Research and Advanced Technology for Digital Libraries, 10th European Conference, ECDL 2006, Alicante, Spain, September 17-22, 2006, Proceedings*, pages 208–219, 2006.

[1461] R. Shen, N. S. Vemuri, W. Fan, and E. A. Fox. Integration of complex archeology digital libraries: An ETANA-DL experience. *Inf. Syst.*, 33(7-8):699–723, 2008.

[1462] X. Shen, B. Tan, and C. Zhai. Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 43–50, 2005.

[1463] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[1464] S. Shi, G. Yang, D. Wang, J. Yu, S. Qu, and M. Chen. Making Peer-to-Peer Keyword Searching Feasible Using Multi-level Partitioning. In *IPTPS'04: Proceedings of the 3rd International workshop on Peer-to-Peer Systems*, La Jolla, CA, USA, 2004.

[1465] N. Shivakumar and H. Garcia-Molina. SCAM: A copy detection mechanism for digital documents. In *Digital Libraries*, 1995.

[1466] N. Shivakumar and H. Garcia-Molina. Building a scalable and accurate copy detection mechanism. In *Proceedings of the 1st ACM International Conference on Digital Libraries*, pages 160–168, Bethesda, MD, USA, March 1996. ACM.

[1467] N. Shivakumar and H. Garcia-Molina. Finding near-replicas of documents and servers on the web. In P. Atzeni, A. O. Mendelzon, and G. Mecca, editors, *WebDB'98*, volume 1590 of *Lecture Notes in Computer Science*, pages 204–212, Valencia, Spain, March 1999. Springer.

[1468] V. Shkapenyuk and T. Suel. Design and implementation of a high-performance distributed Web crawler. In *Proceedings of the 18th International Conference on Data Engineering (ICDE)*, San Jose, California, February 2002. IEEE CS Press.

[1469] D. Shkarin. PPM: One step to practicality. In *Proc. 12th IEEE Data Compression Conference (DCC'02)*, page 202, 2002.

[1470] B. Shneiderman and G. Kearsley. *Hypertext Hands-On! An Introduction to a New Way of Organizing and Accessing Information.* Addison-Wesley Publishing Co., Reading, MA, 1989. includes two disks.

[1471] B. Shneiderman and C. Plaisant. Strategies for evaluating information visualization tools: multi-dimensional in-depth long-term case studies. *Proceedings of the 2006 conference Advanced Visual Interfaces (AVI'04), Workshop on Beyond time and errors: novel evaluation methods for information visualization,* pages 1–7, 2006.

[1472] B. Shneiderman, C. Plaisant, M. Cohen, and S. Jacobs. *Designing the user interface: strategies for effective human-computer interaction, 5/E.* Addison Wesley, 2009.

[1473] M. Shokouhi, J. Zobel, F. Scholer, and S. Tahaghoghi. Capturing collection size for distributed non-cooperative retrieval. In *Proceedings of the Annual ACM SIGIR Conference*, Seattle, WA, USA, August 2006. ACM Press.

[1474] M. Shokouhi, J. Zobel, S. M. Tahaghoghi, and F. Scholer. Using query logs to establish vocabularies in distributed information retrieval. *Information Processing and Management,* 43(1), January 2007.

[1475] L. Si, R. Jin, J. Callan, and P. Olgilvie. A language modeling framework for resource selection and results mergin g. In *Proceedings of the Conference on Information Knowledge Management (CIKM)*, 2002.

[1476] B. Sigurbjornsson, J. Kamps, and M. de Rijke. An element-based approach to XML retrieval. In *Proceedings INEX 2003 Workshop*, pages 19–26, 2004.

[1477] B. Sigurbjörnsson and R. van Zwol. Flickr tag recommendation based on collective knowledge. In *WWW'08: Proceeding of the 17th International Conference on World Wide Web*, pages 327–336, New York, NY, USA, 2008. ACM.

[1478] I. Silva, B. A. Ribeiro-Neto, P. Calado, E. S. de Moura, and N. Ziviani. Link-based and content-based evidential information in a belief network model. In *SIGIR*, pages 96–103, 2000.

[1479] C. Silverstein, M. Henzinger, M. Hannes, and M. Moricz. Analysis of a very large alta vista query log. In *SIGIR Forum*, pages 6–12, 1999. 33(3).

[1480] F. Silvestri. Sorting out the document identifier assignment problem. In *ECIR*, pages 101–112, 2007.

[1481] F. Silvestri. Mining query logs: Turning search usage data into knowledge. *Foundations and Trends in Information Retrieval*, 4(1-2):1–174, 2009.

[1482] F. Silvestri, S. Orlando, and R. Perego. Assigning identifiers to documents to enhance the clustering property of fulltext indexes. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on research and development in information retrieval*, pages 305–312, New York, NY, USA, 2004. ACM Press.

[1483] Sindice: The semantic Web index, 2008. `http://sindice.com`.

[1484] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proc. ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 21–29, Zurich, Switzerland, 1996.

[1485] A. Singhal, M. Mitra, and C. Buckley. Learning routing queries in a query zone. In *SIGIR*, pages 25–32. ACM Press, 1997.

[1486] G. Skobeltsyn and K. Aberer. Distributed Cache Table: Efficient Query-Driven Processing of Multi-Term Queries in P2P Networks. In *P2PIR'06: Proceedings of the workshop on Information Retrieval in Peer-to-Peer Networks*, Arlington, VA, USA, 2006.

[1487] G. Skobeltsyn, F. Junqueira, V. Plachouras, and R. Baeza-Yates. ResIn: A Combination of Result Caching and Index Pruning for High-performance Web Search Engines. In *SIGIR'08: Proceedings of the 31st International ACM SI-*

GIR conference on Research and Development in Information Retrieval, Singapore, 2008.

[1488] G. Skobeltsyn, T. Luu, I. Podnar Žarko, M. Rajman, and K. Aberer. Web Text Retrieval with a P2P Query-Driven Index. In *SIGIR'07: Proceedings of the 30th International ACM SIGIR conference on Research and Development in Information Retrieval*, Amsterdam, The Netherlands, 2007.

[1489] M. Slaney. Mixtures of probability experts for audio retrieval and indexing. In *Proc. 2002 IEEE International Conference on Multimedia and Expo*, volume 1, pages 345–348, 2002.

[1490] M. Slaney and M. Casey. Locality-sensitive hashing for finding nearest neighbors. *IEEE Signal Processing Magazine*, 25(2):128–131, March 2008.

[1491] M. Slaney, D. P. W. Ellis, M. Sandler, M. Goto, and M. Goodwin. *Special Issue on Music Information Retrieval, IEEE Transactions on Audio, Speech and Signal Processing*, volume 16. IEEE, February, 2008.

[1492] M. Slaney and G. McRoberts. BabyEars: A recognition system for affective vocalizations. *Speech Communication*, 39:367–384, 2003.

[1493] M. Slaney, D. Ponceleón, and J. Kaufman. Multimedia edges: Finding hierarchy in all dimensions. In *Proceedings of 9th ACM International Conference on Multimedia*, October 2001.

[1494] M. Slaney and W. White. Similarity based on rating data. In *Proceedings on the International Society of Music-Information Retrieval*, September 2007.

[1495] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and TRECVID. In *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, New York, NY, USA, 2006. ACM Press.

[1496] J. Smith, M. Campbell, M. Naphade, A. Natsev, and J. Tesic. Learning and classification of semantic concepts in broadcast video. Technical report, IBM, 2004.

[1497] M. A. Smith and T. Kanade. Video skimming for quick browsing based on audio and image characterization. Technical Report CMU-CS-95-186 School of Computer Science Tech Report, Carnegie Mellon University, 1995.

[1498] C. G. M. Snoek and M. Worring. Concept-based video retrieval. *Foundations and Trends in Information Retrieval*, 2(4):215–322, 2009.

[1499] R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast—but is it good? evaluating non-expert annotations for natural language tasks. In *EMNLP—Conference on Empirical Methods on Natural Language Processing*, 2008.

[1500] I. Soboroff. Dynamic test collections: measuring search effectiveness on the live web. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 276–283, New York, NY, USA, 2006. ACM.

[1501] D. Soergel. *Indexing Languages and Thesauri: Construction and Maintenance*. Melville Publishing Co., Los Angeles, CA, 1974.

[1502] F. Song and B. Croft. A general language model for information retrieval. In *ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 279–280, 1999.

[1503] R. Song, Z. Luo, J.-R. Wen, Y. Yu, and H.-W. Hon. Identifying ambiguous queries in Web search. In C. L. Williamson, M. E. Zurko, P. F. Patel-Schneider, and P. J. Shenoy, editors, *WWW*, pages 1169–1170, Banff, Alberta, Canada, May 2007. ACM.

[1504] K. Sparck Jones. A statistical interpretation of term specificity and its application to retrieval. *Journal of Documentation*, 28(1):11–20, 1972.

[1505] K. Sparck Jones. Index term weighting. *Information Storage and Retrieval*, 9(11):619–633, 1973.

[1506] K. Sparck Jones. Experiments in relevance weighting of search terms. *Information Processing & Management*, 15(13):133–144, 1979.

[1507] K. Sparck Jones. Search term relevance weighting given little relevance information. *Journal of Documentation*, 35(1):30–48, 1979.

[1508] K. Sparck Jones. The Cranfield Tests. In K. S. Jones, editor, *Information Retrieval Experiment*, pages 256–284. Butterworth, 1981.

[1509] K. Sparck Jones and E. O. Barber. What makes an automatic keyword classification effective. *J. of the American Society for Information Sciences*, 22(3):166–175, 1971.

[1510] K. Sparck Jones and P. Willet. *Readings in Information Retrieval.* Morgan Kaufmann Publishers, Inc., 1997.

[1511] C. Spearman. The proof and measurement of association between two things. *American Journal of Psychology*, 15:72–101, 1904.

[1512] E. Spertus. ParaSite: Mining structural information on the Web. In *6th Int'l WWW Conference*, Santa Clara, CA, USA, April 1997.

[1513] M. Spiliopoulou and L. Faulstich. WUM - A tool for WWW utilization analysis. In *Workshop on Web Databases*, pages 109–115, Valencia, Spain, March 1998.

[1514] D. Spinellis. The decay and failures of Web references. *Communications of the ACM*, 46(1):71–77, January 2003.

[1515] A. Spink and C. Cole, editors. *New Directions in Cognitive Information Retrieval*, volume 29 of *Information Retrieval.* Springer, Netherlands, 2005.

[1516] A. Spink, H. Greisdorf, and J. Bateman. From Highly Relevant to Not Relevant: Examining Different Regions of Relevance. *Information Processing and Management*, 34(5):599–621, 1998.

[1517] A. Spink and B. J. Jansen. *Web Search: Public Searching of the Web.* Information Science and Knowledge Management. Springer, 2004.

[1518] A. Spink, B. J. Jansen, C. Blakely, and S. Koshman. A study of results overlap and uniqueness among major Web search engines. *Information Processing & Management*, 42(5):1379–1391, September 2006.

[1519] A. Spink, B. J. Jansen, D. Wolfram, and T. Saracevic. From e-sex to e-commerce: Web search changes. *Computer*, 35(3):107–109, 2002.

[1520] A. Spink, S. Ozmutlu, H. C. Ozmutlu, and B. J. Jansen. U.S. versus European Web searching trends. *SIGIR Forum*, 26(2), 2002.

[1521] A. Spink, D. Wolfram, M. B. J. Jansen, and T. Saracevic. Searching the web: the public and their queries. *Journal of the American Society for Information Science and Technology*, 52(3):226–234, 2001.

[1522] A. Spink and M. Zimmer. *Web Search: Multidisciplinary Perspectives.* Information Science and Knowledge Management. Springer, 2008.

[1523] J. Spool. *Web Site Usability: A Designer's Guide.* Morgan Kaufmann, 1998.

[1524] J. Spool. Usability beyond common sense, 2002. http://www.bcs-hci.org.uk/talks/Spool/UIE-BeyondCommonSense.pdf.

[1525] S. H. Srinivasan and M. Slaney. A bipartite graph model for associating images and text. In *IJCAI-2007 Workshop on Multimodal Information Retrieval*, 2007.

[1526] P. Srinivasdan. Thesaurus construction. In W. Frakes and R. Baeza-Yates,

editors, *Information Retrieval: Data Structures & Algorithms*, pages 161–218. Prentice Hall, 1992.

[1527] R. M. Stallman. Emacs the extensible, customizable self-documenting display editor. *SIGPLAN Not.*, 16(6):147–156, 1981.

[1528] C. Stanfill. Partitioned posting files: A parallel inverted file structure for information retrieval. In *Proc. 13th Inter. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 413–428, Brussels, Belgium, 1990.

[1529] C. Stanfill. Parallel information retrieval algorithms. In W. B. Frakes and R. Baeza-Yates, editors, *Information Retrieval Data Structures & Algorithms*, pages 459–497. Prentice Hall, Englewood Cliffs, NJ, USA, 1992.

[1530] C. Stanfill and B. Kahle. Parallel free-text search on the Connection Machine system. *Commun. ACM*, 29(12):1229–1239, Dec. 1986.

[1531] C. Stanfill, R. Thau, and D. Waltz. A parallel indexed algorithm for information retrieval. In *Proc. 12th Inter. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 88–97, Cambridge, USA, June 1989.

[1532] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 1999.

[1533] J. G. Steiner, C. Neuman, and J. I. Schiller. Kerberos: An authentication service for open network systems. In *Winter 1988 USENIX Conference*, pages 191–201, Dallas, TX, 1988. USENIX Association.

[1534] R. Steinmetz and K. Nahrstedt. *Multimedia - Computing, Communications and Applications.* Prentice Hall, 1996. 854 pages.

[1535] D. Stenmark. Method for intranet search engine evaluations. In *Proceedings of IRIS22*, Department of CS/IS, University of Jyväskylä, Finland, August 1999. http://w3.informatik.gu.se/~dixi/publ/method.pdf.

[1536] E. Stoica, M. Hearst, and M. Richardson. Automating Creation of Hierarchical Faceted Metadata Structures. In *Human Language Technologies: the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007)*, pages 244-251, 2007.

[1537] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160, New York, NY, USA, 2001. ACM.

[1538] T. Strzalkowski, editor. *Natural Language Information Retrieval.* Kluwer Academic Publishers, 1999.

[1539] A.-J. Su, D. Choffnes, A. Kuzmanovic, and F. Bustamante. Drafting behind Akamai (travelocity-based detouring). In *Proceedings of the ACM SIGCOMM Conference*, pages 435–446, Pisa, Italy, September 2006.

[1540] Q. Su, D. Pavlov, J. Chow, and W. Baker. Internet-scale collection of human-reviewed data. In *WWW'07: Proc. of the International World Wide Web Conference*, 2007.

[1541] T. Suel, C. Mathur, J.-W. Wu, J. Zhang, A. Delis, M. Kharrazi, X. Long, and K. Shanmugasundaram. ODISSEA: A Peer-to-Peer Architecture for Scalable Web Search and Information Retrieval. In *WebDB'03: Proceedings of the International workshop on Web and Databases*, San Diego, CA, USA, 2003.

[1542] H. Suleman, A. Atkins, M. A. Gonçalves, R. K. France, E. A. Fox, V. Chachra, and M. Crowder. Networked digital library of theses and dissertations: Bridging the gaps for global access - part 1. *D-Lib Magazine*, 7(8), 2001.

[1543] D. Sullivan. Search Engine Watch. http://www.searchenginewatch.com, 1997.

[1544] T. Sumner, M. Khoo, M. Recker, and M. Marlino. Understanding educator perceptions of 'quality' in digital libraries. In *Proc. of JCDL'03*, pages 269–

279, 2003.

[1545] D. Sunday. A very fast substring search algorithm. *Communications of the ACM*, 33(8):132–142, Aug. 1990.

[1546] J. Surowiecki. *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations.* Random House, 2004.

[1547] A. Sutcliffe and M. Ennis. Towards a cognitive theory of information retrieval. *Interacting with Computers*, 10:321–351, 1998.

[1548] R. Swan and J. Allan. Aspect Windows, 3-D Visualizations, and Indirect Comparisons of Information Retrieval Systems. *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'98)*, pages 173–181, 1998.

[1549] Swish++. http://homepage.mac.com/pauljlucas/software/swish/, 2007.

[1550] Swish-e. http://www.swish-e.org/, 2007.

[1551] D. Tabatabai and B. Shore. How experts and novices search the Web. *Library & Information Science Research*, 27(2):222–248, 2005.

[1552] J. Tague-Sutcliffe. Measuring the informativeness of a retrieval process. In *Proc of the Fifteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 23–36, Denmark, 1992.

[1553] V. Tahani. A fuzzy model of document retrieval systems. *Information Processing & Management*, 12:177–187, 1976.

[1554] J. I. Tait, editor. *Charting a New Course: Natural Language Processing and Information Retrieval. Essays in Honour of Karen Spărck Jones.* Springer, 2005.

[1555] H. Tamura, S. Mori, and T. Yamawaki. Texture features corresponding to visual perception. *IEEE Transactions on System, Man and Cybernatic*, 6, 1978.

[1556] B. Tan, X. Shen, and C. Zhai. Mining long-term search history to improve search accuracy. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 718–723, New York, NY, USA, 2006. ACM.

[1557] P. N. Tan and V. Kumar. Discovery of Web robots session based on their navigational patterns. *Data Mining and Knowledge discovery*, 6(1):9–35, 2002.

[1558] C. Tang, Z. Xu, and S. Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 175–186, New York, NY, USA, 2003. ACM.

[1559] C. Tang, Z. Xu, and M. Mahalingam. pSearch: Information retrieval in structured overlays. *SIGCOMM Comput. Commun. Rev.*, 33(1):89–94, 2003.

[1560] Y. Taniguchi. An intuitive and efficient access interface to real-time incoming video based on automatic indexing. In *Proc. ACM Multimedia*, pages 25–33, November 1995.

[1561] Y. Taniguchi, A. Akutsu, and Y. Tonomura. PanoramaExcerpts: Extracting and packing panoramas for video browsing. In *MULTIMEDIA '97: Proceedings of the Fifth ACM International Conference on Multimedia*, pages 427–436, New York, NY, USA, Nov 1997. ACM.

[1562] R. Tansley, M. Bass, D. Stuve, M. Branschofsky, D. Chudnov, G. McClellan, and M. Smith. DSpace: An institutional digital repository system. In *Proc. of the 3rd Joint Conference on Digital Libraries*, pages 87–97, Houston, Texas, 2003.

[1563] T. Tao and C. Zhai. An exploration of proximity measures in information re-

trieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'07)*, pages 295–302. ACM Press, 2007.

[1564] C. M. Taskiran, Z. Pizlo, A. Amir, D. B. Ponceleón, and E. J. Delp. Automated video program summarization using speech transcripts. *IEEE Transactions in Multimedia*, 8(4):775–791, 2006.

[1565] S. Tauro, C. Palmer, G. Siganos, and M. Faloutsos. A simple conceptual model for the internet topology. In *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, volume 3, pages 1667–1671, 2001.

[1566] M. Taylor, J. Guiver, S. Robertson, and T. Minka. Softrank: optimizing non-smooth rank metrics. In *WSDM '08: Proceedings of the international conference on Web search and Web data mining*, pages 77–86, Palo Alto, California, USA, 2008. ACM Press.

[1567] E. S. Team. Eprints services, 2006. http://www.eprints.org/services/.

[1568] J. Teevan, E. Adar, R. Jones, and M. A. S. Potts. Information Re-retrieval: Repeat Queries in Yahoo's Logs. In *SIGIR'07: Proceedings of the 30th International ACM SIGIR conference on Research and Development in Information Retrieval*, Amsterdam, The Netherlands, 2007.

[1569] J. Teevan, C. Alvarado, M. Ackerman, and D. Karger. The perfect search engine is not enough: a study of orienteering behavior in directed search. *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'04)*, pages 415–422, 2004.

[1570] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proceedings of ACM SIGIR '05*, pages 449–456, New York, NY, USA, 2005. ACM.

[1571] J. Teevan, S. T. Dumais, and E. Horvitz. Characterizing the value of personalizing search. In *Proceedings of ACM SIGIR '07*, pages 757–758, New York, NY, USA, 2007. ACM.

[1572] TEI. A gentle introduction to SGML. Technical report, Text Encoding Initiative, 1996. http://www.sil.org/sgml/gentle.html.

[1573] L. Teodosio and W. Bender. Salient stills. *ACM Trans. Multimedia Comput. Commun. Appl.*, 1(1):16–36, 2005.

[1574] Terrier. http://ir.dcs.gla.ac.uk/terrier/, 2007.

[1575] M. Thelwall. *Link Analysis: An Information Science Approach.* Academic Press, December 2004.

[1576] M. Thelwall and D. Wilkinson. Graph structure in three national academic webs: Power laws with anomalies. *Journal of the American Society for Information Science and Technology*, 54(8):706–712, 2003.

[1577] A. Theobald and G. Weikum. The Index-Based XXL Search Engine for Querying XML Data with Relevance Ranking. In *EDBT*, pages 477–495, 2002.

[1578] M. Theobald, H. Bast, D. Majumdar, R., and G. Weikum. TopX: efficient and versatile top- query processing for semistructured data. *VLDB Journal*, 17(1):81–115, 2008.

[1579] M. Theobald, R. Schenkel, and G. Weikum. TopX and XXL at INEX 2005. In *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005*, pages 282–295, Dagstuhl Castle, Germany, 2006. Revised Selected Papers.

[1580] P. Thomas. *Server characterisation and selection for personal metasearch.* PhD thesis, Australian National University, 2008. http://es.csiro.au/pubs/thomas_thesis.pdf.

[1581] P. Thomas and D. Hawking. Evaluation by comparing result sets in context. In *ACM Int Conference on Information and Knowledge Management (CIKM)*, pages 94–101, 2006.

[1582] P. Thomas and D. Hawking. Evaluating sampling methods for uncooperative collections. In *Proceedings of ACM SIGIR 2007*, pages 503–510, July 2007. http://david-hawking.net/pubs/fp347-thomas.pdf.

[1583] P. Thomas and D. Hawking. Experiences evaluating personal metasearch. In *Proceedings of IIiX*, London, 2008. http://es.csiro.au/pubs/thomas_iiix08.pdf.

[1584] K. Thompson. Regular expression search algorithm. *Communications of ACM*, 11:419–422, 1968.

[1585] K. M. Ting and I. H. Witten. Stacked generalizations: When does it work? In *IJCAI (2)*, pages 866–873, 1997.

[1586] H. Tirri. Search in vain: Challenges for Internet search. *Computer*, 36(1):115–116, 2003.

[1587] TodoCL, 2000. http://www.todocl.com.

[1588] A. Tomasic and H. García-Molina. Caching and database scaling in distributed shared-nothing information retrieval systems. In *Proc. of the ACM SIGMOD Inter. Conf. on Management of Data*, pages 129–138, Washington, D.C., USA, May 1993.

[1589] A. Tomasic and H. Garcia-Molina. Performance of inverted indices in shared-nothing distributed text document information retrieval systems. In *Proceedings of the second international conference on Parallel and distributed information systems*, pages 8–17, San Diego, California, United States, 1993. IEEE Computer Society Press.

[1590] A. Tomasic and H. García-Molina. Performance issues in distributed shared-nothing information retrieval systems. *Inf. Process. & Mgmnt.*, 32(6):647–665, 1996.

[1591] A. Tombros, B. Larsen, and S. Malik. The interactive track at INEX 2004. In *Advances in XML Information Retrieval, Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*, pages 410–423, Dagstuhl Castle, Germany, 2005. Revised Selected Papers.

[1592] A. Tombros, S. Malik, and B. Larsen. Report on the INEX 2004 interactive track. *SIGIR Forum*, 39(1):43–49, 2005.

[1593] A. Tombros and M. Sanderson. Advantages of query biased summaries in information retrieval. *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'98)*, pages 2–10, 1998.

[1594] The TREC NIST site, 2008. http://trec.nist.gov.

[1595] A. Trotman. Learning to rank. *Information Retrieval*, 8(3):359–381, 2005.

[1596] A. Trotman. Narrowed Extended XPath I (NEXI). In *Encyclopedia of Database Systems*. Springer, 2009.

[1597] A. Trotman. Processing structural constraints. In *Encyclopedia of Database Systems*. Springer, 2009.

[1598] A. Trotman and S. Geva. Report on the SIGIR 2006 workshop on XML element retrieval methodology. *SIGIR Forum*, 40(2):42–48, 2006.

[1599] A. Trotman, S. Geva, and J. Kamps. Report on the SIGIR 2007 workshop on focused retrieval. *SIGIR Forum*, 41(2):97–103, 2007.

[1600] A. Trotman and M. Lalmas. Report on the INEX 2005 workshop on element retrieval methodology. *SIGIR Forum*, 39(2):46–51, 2005.

[1601] A. Trotman and M. Lalmas. Why structural hints in queries do not help XML retrieval. In *29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA*, pages 711–712. 2006.

[1602] A. Trotman and B. Sigurbjornsson. Narrowed Extended XPath I (NEXI). In *Advances in XML Information Retrieval, Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*, pages 16–40, Dagstuhl Castle, Germany, 2005. Revised Selected Papers.

[1603] M.-F. Tsai, T.-Y. Liu, T. Qin, H.-H. Chen, and W.-Y. Ma. Frank: a ranking method with fidelity loss. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 383–390, Amsterdam, The Netherlands, 2007. ACM Press.

[1604] T. Tsikrika. Aggregation-based Structured Text Retrieval. In *Encyclopedia of Database Systems*. Springer, 2009.

[1605] E. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Chelshire, CT, 1983.

[1606] E. Tufte. Beautiful Evidence. *Information Design Journal*, 15(2):188–191, 2007.

[1607] D. Tunkelang. *Faceted Search*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan Claypool, 2009.

[1608] M. Turk. A Random Walk through Eigenspace. *IEICE Transactions on Information and Systems*, Vol. E84-D(12):1586–1595, December 2001.

[1609] H. Turtle and W. B. Croft. Inference networks for document retrieval. In *Proceedings of the Thirteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Information Retrieval Models (1), pages 1–24, 1990.

[1610] H. Turtle and W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3):187–222, July 1991.

[1611] H. R. Turtle. *Inference Networks for Document Retrieval*. PhD thesis, University of Massachusetts at Amherst, Department of Computer Science, February 1991.

[1612] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: a survey. *Found. Trends. Comput. Graph. Vis.*, 3(3):177–280, 2008.

[1613] S. Uchihashi, J. Foote, A. Girgensohn, and J. Boreczky. Video manga: generating semantically meaningful video summaries. In *MULTIMEDIA '99: Proceedings of the Seventh ACM International Conference on Multimedia (Part 1)*, pages 383–392, New York, NY, USA, 1999. ACM.

[1614] E. Ukkonen. Finding approximate patterns in strings. *Journal of Algorithms*, 6:132–137, 1985.

[1615] E. Ukkonen. Approximate string matching over suffix trees. In A. Apostolico, M. Crochemore, Z. Galil, and U. Manber, editors, *Proc. of Combinatorial Pattern Matching*, number 684 in LNCS, pages 228–242, Padova, Italy, 1993. Springer-Verlag.

[1616] E. Ukkonen. Constructing suffix trees on-line in linear time. *Algorithmica*, 14(3):249–260, Sep 1995.

[1617] Unicode Consortium, Unicode. http://www.unicode.org/.

[1618] University of California Libraries. Bibliographic Services Task Force. Rethinking how we provide bibliographic services for the university of california. Final report, University of California, December 2005. Available at http://libraries.universityofcalifornia.edu/sopag/BSTF/Final.pdf.

[1619] T. Upstill, N. Craswell, and D. Hawking. Query-independent evidence in home page finding. *ACM Transactions on Information Systems (TOIS)*, 21(3):286–313, 2003. http://es.csiro.au/pubs/upstill_tois03.pdf.

[1620] P. Vakkari. Relevance and Contributing Information Types of Searched Documents in Task Performance. *Proceedings of the 23th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'00)*, pages 2–9, 2000.

[1621] H. van de Sompel, J. A. Young, and T. B. Hickey. Using the OAI-PMH ... differently. *D-Lib Magazine*, 9(7/8), July/Aug. 2003.

[1622] T. P. van der Weide, T. W. C. Huibers, and P. van Bommel. The incremental searcher satisfaction model for information retrieval. *The Computer Journal*, 41(5):311–318, 1998.

[1623] A. van Deursen, P. Klint, and J. Visser. Domain-specific languages: An annotated bibliography. *ACM SIGPLAN Notices*, 35(6):26–36, June 2000.

[1624] C. van Rijsbergen. *Information Retrieval*. Butterwords, 1979.

[1625] C. van Rijsbergen. *The Geometry of Information Retrieval*. Cambridge University Press, August 2004.

[1626] R. van Zwol. B^3-SDR and Effective Use of Structural Hints. In *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2005*, pages 146–160, Dagstuhl Castle, Germany, 2006. Revised Selected Papers.

[1627] R. van Zwol, J. Baas, H. van Oostendorp, and F. Wiering. Bricks: the Building Blocks to Tackle Query Formulation in Structured Document Retrieval. In *Advances in Information Retrieval, 28th European Conference on IR Research*, pages 314–325, London, UK, 2006.

[1628] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, NY, 1998.

[1629] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *Proceedings of the IEEE International Conference on Computer Vision*, Rio de Janeiro, Brazil, October 2007.

[1630] A. Veerasamy and N. Belkin. Evaluation of a tool for visualization of information retrieval results. *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'96)*, pages 85–92, 1996.

[1631] L. Veiga e Silva, M. A. Gonçalves, and A. H. F. Laender. Evaluating a digital library self-archiving service: the bdbcomp user case study. *Information Processing & Management*, 43(4), 2007.

[1632] A. Veloso, H. M. de Almeida, M. A. Gonçalves, and W. Meira Jr. Learning to rank at query-time using association rules. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–274, Singapore, July 2008.

[1633] A. Veloso, W. Meira Jr., M. Cristo, M. A. Gonçalves, and M. J. Zaki. Multi-evidence, multi-criteria, lazy associative document classification. In P. S. Yu, V. J. Tsotras, E. A. Fox, and B. Liu, editors, *Proceedings of the 2006 ACM CIKM International Conference on Information and Knowledge Management, Arlington, Virginia, USA, November 6-11, 2006*, pages 218–227. ACM Press, 2006.

[1634] A. Veloso, W. Meira Jr., M. A. Gonçalves, and M. J. Zaki. Multi-label lazy associative classification. In *Knowledge Discovery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007, Proceedings*, pages 605–612. Springer, 2007.

[1635] N. S. Vemuri, R. da Silva Torres, R. Shen, M. A. Gonçalves, W. Fan, and E. A. Fox. A content-based image retrieval service for archaeology collections. In *Research and Advanced Technology for Digital Libraries, 10th European Conference, ECDL 2006, Alicante, Spain, September 17-22, 2006, Proceedings*, pages 438–440. Springer, 2006.

[1636] J. Verhoeff, W. Goffmann, and J. Belzer. Inefficiency of the use of Boolean functions for information retrieval systems. *Communications of the ACM*, 4(12):557–558, 594, Dec. 1961.

[1637] F. Viégas, M. Wattenberg, F. van Ham, J. Kriss, and M. McKeon. Many Eyes:

A Site for Visualization at Internet Scale. *IEEE Transactions on Visualization and Computer Graphics*, pages 1121–1128, 2007.

[1638] M. V. Vieira, B. M. Fonseca, R. Damazio, P. B. Golgher, D. C. Reis, and B. A. Ribeiro-Neto. Efficient search ranking in social networks. In *CIKM*, pages 563–572, 2007.

[1639] R. C. Vieira, P. Calado, A. S. da Silva, A. H. F. Laender, and B. A. Ribeiro-Neto. Structuring keyword-based queries for Web databases. In *JCDL*, pages 94–95, 2002.

[1640] C. L. Viles and J. C. French. Dissemination of collection wide information in a distributed information retrieval system. In *Proc. 18th Inter. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 12–20, Seattle, WA, USA, July 1995.

[1641] V. Vinay, I. J. Cox, N. Milic-Frayling, and K. Wood. On ranking the effectiveness of searches. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 398–404, Seattle, Washington, USA, 2006.

[1642] J.-N. Vittaut and P. Gallinari. Machine learning ranking for structured information retrieval. In *Advances in Information Retrieval, 28th European Conference on IR Research, ECIR 2006, London, UK, 2006,*, pages 338–349, 2006.

[1643] Vivísimo, 1996. http://www.vivisimo.com.

[1644] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *CHI '04: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 319–326, New York, NY, USA, 2004. ACM Press.

[1645] E. Voorhees. *The Effectiveness and Efficiency of Agglomerative Hierarchic Clustering in Document Retrieval.* PhD thesis, Cornell University, 1986.

[1646] E. Voorhees. The TREC-8 Question Answering track report. In *TREC-8: Proceedings of the Eighth Text Retrieval Conference*, pages 77–82, 2000.

[1647] E. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing & Management*, 36(5):697–716, 2000.

[1648] E. Voorhees. The philosophy of information retrieval evaluation. In *Evaluation of Cross-Language Information Retrieval Systems*, pages 143–170. Springer Verlag / Heidelberg, 2002. Lecture Notes in Computer Science.

[1649] E. Voorhees. Overview of the TREC 2006. In *Proc. of the Fifteenth Text REtrieval Conference.* NIST Special Publication, Gaithersburg, MD, USA, 2006.

[1650] E. Voorhees. Overview of TREC 2007. In *16th Text Retrieval Conference (TREC)*, 2007.

[1651] E. Voorhees and D. Harman. Overview of the sixth text retrieval conference (TREC-6). In E. Voorhees and D. Harman, editors, *Proceedings of the Sixth Text REtrieval Conference (TREC-6).* NIST Special Publication, 1997.

[1652] E. M. Voorhees, N. K. Gupta, and B. Johnson-Laird. The collection fusion problem. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, pages 95–104, Gaithersburg, MD, USA, 1995. Dept. of Commerce, National Institute of Standards and Technology. Special Publication 500-226.

[1653] E. M. Voorhees, N. K. Gupta, and B. Johnson-Laird. Learning collection fusion strategies. In *Proceedings of ACM SIGIR'95*, pages 172–179, 1995.

[1654] E. M. Voorhees and D. K. Harman. *TREC: Experiment and Evaluation in Information Retrieval.* MIT Press, Cambridge, Mass., USA, 2005.

[1655] W3C. Extensible markup language (XML) 1.0. Technical report, WWW Consortium (W3C), 1998. http://www.w3.org/TR/1998/REC-xml-19980210.

[1656] W3C. HTML 4.0 specification. Technical report, WWW Consortium (W3C), 1998. http://www.w3.org/TR/1998/REC-html40-19980424/.

[1657] W3C. XML linking language (XLink). Technical report, WWW Consortium (W3C), 1998. http://www.w3.org/TR/1998/WD-xlink-19980303.

[1658] W3C. XSL requirements summary. Technical report, WWW Consortium (W3C), 1998. http://www.w3.org/TR/1998/WD-XSLReq-19980511.

[1659] W3C. XML Schema. http://www.w3.org/XML/Schema, 2001.

[1660] W3C. Resource Description Framework (RDF), 2004. http://www.w3.org/RDF.

[1661] W3C. SPARQL, 2008. http://www.w3.org/TR/rdf-sparql-query/.

[1662] A. Waern. User involvement in automatic filtering: An experimental study. *User Modeling and User-Adapted Interaction (UMUAI)*, 14(2–3):201–237, 2001.

[1663] R. Wan. *Browsing and Searching Compressed Documents.* PhD thesis, Department of Computer Science and Software Engineering, University of Melbourne, Melbourne, Australia, 2003.

[1664] A. Wang. An industrial strength audio search algorithm. In *ISMIR*, 2003.

[1665] D. Wang and G. J. Brown. *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications.* Wiley-IEEE Press, September 2006.

[1666] S. Warner. Eprints and the open archives initiative. *CoRR*, cs.DL/0307008, 2003.

[1667] S. Wartick. Boolean operations. In W. Frakes and R. Baeza-Yates, editors, *Information Retrieval: Data Structures & Algorithms*, pages 264–292. Prentice Hall, 1992.

[1668] D. J. Waters. What are digital libraries. *CLIR issues*, (4), July/August 1998. http://www.clir.org/pubs/issues/issues04.html#dlf.

[1669] M. Wattenberg and B. Fernanda. The Word Tree, an Interactive Visual Concordance. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1221–1228, 2008.

[1670] M. Wattenberg and J. Kriss. Designing for Social Data Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):549–557, 2006.

[1671] A. Waugh, R. Wilkinson, B. Hills, and J. Dell'oro. Preserving digital information forever. In *DL'00: Proceedings of the 5th ACM International Conference on Digital Libraries*, pages 175–184, San Antonio, Texas, 2000.

[1672] J. Weatherley, T. Sumner, M. Khoo, M. Wright, and M. Hoffmann. Partnership reviewing: a cooperative approach for peer review of complex educational resources. In *Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 106–114, Portland, Oregon, 2002.

[1673] W. Webber, A. Moffat, J. Zobel, and R. Baeza-Yates. A Pipelined Architecture for Distributed Text Query Evaluation. *Information Retrieval*, 10(3), 2007.

[1674] Webglimpse. http://www.webglimpse.net/, 2007.

[1675] S. Weibel and E. Miller. Dublin Core Metadata, 1997. http://purl.org/metadata/dublin_core.

[1676] A. Weigend, E. Wiener, and J. Pedersen. Exploiting hierarchy in text categorization. *Information Retrieval*, 1(3):193–216, 1999.

[1677] K. Weinberger, M. Slaney, and R. van Zwol. Resolving tag ambiguity. In

MULTIMEDIA '08: Proceedings of the 16th International Conference on Multimedia, New York, NY, USA, 2008. ACM.

[1678] P. Weiner. Linear pattern matching algorithms. In *Proc. IEEE Symp. on Switching and Automata Theory*, pages 1–11, 1973.

[1679] R. Weiss, B. Vélez, M. Sheldon, C. Nemprempre, P. Szilagyi, and D. Gifford. HyPursuit: A hierarchical network engine that exploits content-link hypertext clustering. In *7th ACM Conference on Hypertext and Hypermedia*, pages 180–193, Washington, D.C., USA, 1996.

[1680] J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *ESANN*, pages 219–224, 1999.

[1681] R. White, M. Bilenko, and S. Cucerzan. Studying the Use of Popular Destinations to Enhance Web Search Interaction. *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'07)*, 2007.

[1682] R. White, J. Jose, and I. Ruthven. A task-oriented study on the influencing effects of query-biased summarisation in Web searching. *Information Processing and Management*, 39(5):707–733, 2003.

[1683] R. White, J. Jose, and I. Ruthven. Using Top-Ranking Sentences for Web Search Result Presentation. *Proceedings of the 12th International Conference on World Wide Web (WWW'03)*, 2003.

[1684] R. White and G. Marchionini. Examining the effectiveness of real-time query expansion. *Information Processing and Management*, 43(3), 2007.

[1685] R. White and D. Morris. Investigating the querying and browsing behavior of advanced search engine users. *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and development in information retrieval (SIGIR'07)*, 2007.

[1686] R. W. White and R. A. Roth. *Exploratory Search: Beyond the Query-Response Paradigm.* Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan Claypool, 2009.

[1687] R. W. White, I. Ruthven, and J. M. Jose. A study of factors affecting the utility of implicit relevance feedback. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 35–42, New York, NY, USA, 2005. ACM.

[1688] M. Whiting and N. Cramer. WebTheme: Understanding Web Information Through Visual Analytics. In *Proceedings of the First International Semantic Web Conference (ISWC'02)*, pages 460–468. Springer-Verlag London, UK, 2002.

[1689] Wikipedia. http://www.wikipedia.org/, 2001.

[1690] Wikipedia, the Free Encyclopedia. The mother of all demos, September 2006. http://en.wikipedia.org/wiki/The_Mother_of_All_Demos.

[1691] Wikipedia, the Free Encyclopedia. Ted Nelson, September 2006. http://en.wikipedia.org/wiki/Ted_Nelson.

[1692] Wikipedia, the Free Encyclopedia. Information retrieval, 2009. http://en.wikipedia.org/wiki/Information_retrieval.

[1693] Wikipedia, the Free Encyclopedia. Pride & Prejudice (film 2005), 2009. http://en.wikipedia.org/wiki/Pride_&_Prejudice_2005_film.

[1694] Wikipedia, the Free Encyclopedia. Pride and Prejudice, 2009. http://en.wikipedia.org/wiki/Pride_and_Prejudice.

[1695] B. M. Wildemuth, G. Marchionini, M. Yang, G. Geisler, T. Wilkens, A. Hughes, and R. Gruss. How fast is too fast? Evaluating fast forward surrogates for

digital video. In *JCDL*, pages 221–230, 2003.

[1696] R. Wilkinson. Effective retrieval of structured documents. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 311 317. Springer-Verlag New York, Inc., 1994.

[1697] R. Wilkinson and P. Hingston. Using the cosine measure in a neural network for document retrieval. In *Proc. ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 202–210, 1991.

[1698] J. Williams. *Bots and other Internet beasts.* Prentice Hall, 1996.

[1699] W. Willinger and V. Paxson. Where mathematics meets the Internet. *Notices of the AMS*, 45(8):961–970, 1998.

[1700] J. A. Wise, J. J. Thomas, K. Pennock, D. Lantrip, M. Pottier, and A. Schur. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. In *Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'95)*, pages 51–58. IEEE Computer Society Press, 1995.

[1701] I. Witten, D. Bainbridge, G. Paynter, and S. Boddie. The greenstone plugin architecture. In *Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 285–286, Portland, Oregon, 2002.

[1702] I. Witten, H. Boddie, J. Stefan, D. Bainbridge, and R. J. McNab. Greenstone: A comprehensive open-source digital library software system. In *Proceedings of the 5th ACM International Conference on Digital Libraries*, pages 113–121, San Antonio, Texas, 2000.

[1703] I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images.* Van Nostrand Reinhold, New York, 1994.

[1704] I. H. Witten and D. Bainbridge. *How to Build a Digital Library.* Morgan Kaufmann, 2003.

[1705] I. H. Witten, D. Bainbridge, and S. J. Boddie. Power to the people: End-user building of digital library collections. In *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries*, Tools for Constructing and Using Digital Libraries, pages 94–103, Roanoke, VA, 2001.

[1706] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, October 1999.

[1707] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann, San Francisco, 2 edition, 2005.

[1708] I. H. Witten, M. Gori, and T. Numerico. *Web Dragons: Inside the Myths of Search Engine Technology.* Morgan Kaufmann, 2006.

[1709] I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes - Compressing and Indexing Documents and Images.* Morgan Kaufmann Publishers, Inc, San Francisco, CA, second edition, 1999.

[1710] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.

[1711] K. Wittenburg and E. Sigman. Integration of browsing, searching, and filtering in an applet for Web information access. In *Proc. of the ACM Conference on Human Factors in Computing Systems, Late Breaking Track*, Atlanta, GA, USA, 1997. http://www1.acm.org:82/sigs/sigchi/chi97/proceedings/short-talk/kw.htm.

[1712] J. L. Wolf, M. S. Squillante, P. S. Yu, J. Sethuraman, and L. Ozsen. Optimal crawling strategies for Web search engines. In *WWW'02: Proceedings of the 11th international conference on World Wide Web*, pages 136–147, New York, NY, USA, 2002. ACM Press.

[1713] D. Wolfram. A query-level examination of end user searching behaviour on the excite search engine. In *Proceedings of the 28th Annual Conference Canadian Association for Information Science*, 2000.

[1714] D. Wolfram, P. Wang, and J. Zhang. Identifying Web search session patterns using cluster analysis: A comparison of three search environments. *JASIST*, 60(5):896–910, 2009.

[1715] A. Wolman, G. M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. Levy. On the scale and performance of cooperative Web proxy caching. *ACM Operating Systems Review*, 34(5):16–31, December 1999.

[1716] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

[1717] S. Wong, W. Ziarko, V. Raghavan, and P. Wong. On modeling of information retrieval concepts in vector spaces. *ACM Transactions on Database Systems*, 12(2):299–321, 1987.

[1718] S. K. M. Wong, W. Ziarko, and P. C. N. Wong. Generalized vector space model in information retrieval. In *Proc. Eighth ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 18–25, New York, 1985.

[1719] A. Woodruff, P. Aoki, E. Brewer, P. Gauthier, and L. Rowe. An investigation of documents from the World Wide Web. In *5th WWW Conf.*, Paris, France, 1996.

[1720] A. Woodruff, A. Faulring, R. Rosenholtz, J. Morrison, and P. Pirolli. Using thumbnails to search the Web. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'01)*, pages 198–205, 2001.

[1721] K. Woods, W. P. Kegelmeyer, and K. W. Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transsactions on Pattern Analysis and Machine Intellegence*, 19(4):405–410, 1997.

[1722] H. Wu and G. Salton. The estimation of term relevance weights using relevance feedback. *Journal of Documentation*, 37(4):194–214, 1981.

[1723] S. Wu and U. Manber. Agrep – a fast approximate pattern-matching tool. In *Proc. of USENIX Technical Conference*, pages 153–162, 1992.

[1724] S. Wu and U. Manber. Fast text searching allowing errors. *Communications of the ACM*, 35(10):83–91, Oct. 1992.

[1725] Xapian code library. http://www.xapian.org/, 2007.

[1726] F. Xia, T. Y. Liu, J. Wang, W. Zhang, and H. Li. Listwise approach to learning to rank: theory and algorithm. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, pages 1192–1199, New York, NY, USA, 2008. ACM Press.

[1727] Y. Xie and D. R. O'Hallaron. Locality in search engine queries and its implications for caching. In *INFOCOM*, 2002.

[1728] L. Xiong and E. Agichtein. Towards privacy preserving query log publishing and analysis. In *Query Log Analysis Workshop, in conjunction with International Conference on World Wide Web (WWW)*, 2007.

[1729] Z. Xiong, R. Radhakrishnan, A. Divakaran, Y. Rui, and T. S. Huang. *A Unified Framework for Video Summarization, Browsing and Retrieval: With Applications to Consumer and Surveillance Video*. Elsevier, Amsterdam, 2006.

[1730] J. Xu and J. P. Callan. Effective retrieval with distributed collections. In *SIGIR*, pages 112–120, Melbourne, Australia, August 1998. ACM.

[1731] J. Xu and B. Croft. Cluster-based Language Models for Distributed Retrieval. In *SIGIR'99: Proceedings of the 22nd International ACM SIGIR conference on Research and Development in Information Retrieval*, Berkeley, CA, USA, 1999.

[1732] J. Xu and W. Croft. Query expansion using local and global document analysis.

In *Proc. ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 4–11, Zurich, Switzerland, 1996.

[1733] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 391–398, New York, NY, USA, 2007. ACM Press.

[1734] G.-R. Xue, H.-J. Zeng, Z. Chen, Y. Yu, W.-Y. Ma, W. Xi, and W. Fan. Optimizing Web search using web click-through data. In *Proceedings of ACM CIKM '04*, pages 118–126, 2004.

[1735] Yahoo. Searchmonkey, 2008. http://developer.yahoo.com/searchmonkey/.

[1736] Yahoo! Search tips. http://help.yahoo.com/l/us/yahoo/search/basics/basics-04.html, 2009.

[1737] Yahoo! directory: http://search.yahoo.com/dir, 2009.

[1738] H. Yan, S. Ding, and T. Suel. Compressing term positions in Web indexes. In J. Allan, J. A. Aslam, M. Sanderson, C. Zhai, and J. Zobel, editors, *SIGIR*, pages 147–154, Boston, MA, USA, July 2009. ACM.

[1739] H. Yan, S. Ding, and T. Suel. Inverted index compression and query processing with optimized document ordering. In J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, editors, *WWW*, pages 401–410, Madrid, Spain, April 2009. ACM.

[1740] B. Yang and G. Jeh. Retroactive answering of search queries. In *WWW'06: Proceedings of the 15th international conference on World Wide Web*, pages 457–466, New York, NY, USA, 2006. ACM.

[1741] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 13–22, Dublin, Ireland, July 1994.

[1742] Y. Yang, T. Ault, and T. Pierce. Combining multiple learning strategies for effective cross validation. In *Proc. 17th International Conf. on Machine Learning*, pages 1167–1174. Morgan Kaufmann, San Francisco, CA, 2000.

[1743] Y. Yang and X. Liu. A re-examination of text categorization methods. In M. A. Hearst, F. Gey, and R. Tong, editors, *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, pages 42–49, Berkeley, US, 1999.

[1744] Y. Yang and J. Pedersen. A comparative study on feature selection in text categorization. In *Proc. of the 14th International Conference on Machine Learning - ICML-97*, pages 412–420, Nashville, TN, 1997.

[1745] D. Z. Yazti and M. D. Dikaiakos. Design and implementation of a distributed crawler and filtering processor. In *Proceedings of the fifth Next Generation Information Technologies and Systems (NGITS)*, volume 2382 of *Lecture Notes in Computer Science*, pages 58–74, Caesarea, Israel, June 2002. Springer.

[1746] K.-P. Yee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'03)*, pages 401–408. ACM New York, NY, USA, 2003.

[1747] B.-L. Yeo and M. M. Yeung. Retrieving and visualizing video. *Commun. ACM*, 40(12):43–52, 1997.

[1748] B.-L. Yeo and M. M. Yeung. Classification, simplification and dynamic visualization of scene transition graphs for video browsing. *IS&T/SPIE Electronic Imaging 98: Storage and Retrieval for Image and Video Databases VI*, pages 60–70, 1998.

[1749] W. Yih, J. Goodman, and V. R. Carvalho. Finding advertising keywords on Web pages. In L. Carr, D. D. Roure, A. Iyengar, C. A. Goble, and M. Dahlin, editors, *WWW*, pages 213–222, Edinburgh, Scotland, UK, 2006. ACM.

[1750] O. Yilmazel, Finneran, C. M., Liddy, and E. D. Metaextract: an NLP system to automatically assign metadata. In *JCDL'04: Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 241–242, 2004.

[1751] O. Yitzhak, N. Golbandi, N. Harel, R. Lempel, A. Neumann, S. Koifman, D. Sheinwald, E. Shekita, B. Sznajder, and S. Yogev. Beyond basic faceted search. In *WSDM '08: Proceedings of the international conference on Web search and Web data mining*, pages 33–44. ACM, 2008.

[1752] E. Yom-Tov, D. Carmel, A. Darlow, D. Pelleg, S. Errera-Yaakov, and S. Fine. Juru at TREC 2005: Query prediction in the terabyte and the robust tracks. In *TREC 2005*, 2005.

[1753] E. Yom-Tov, S. Fine, D. Carmel, and A. Darlow. Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 512–519, 2005.

[1754] Z. B. Yossef, A. Z. Broder, R. Kumar, and A. Tomkins. Sic transit gloria telae: towards an understanding of the web's decay. In *Proceedings of the 13th conference on World Wide Web*, New York, NY, USA, May 2004. ACM Press.

[1755] D. Young and B. Shneiderman. A graphical filter/flow model for Boolean queries: An implementation and experiment. *Journal of the American Society for Information Science*, 44(6):327–339, July 1993.

[1756] C. T. Yu and G. Salton. Precision weighting–an effective automatic indexing method. *Journal of the ACM*, 23(1):76–88, Jan. 1976.

[1757] H. Yu and M. Young. The impact of Web search engines on subject searching in OPAC. *Information Technology and Libraries*, 23(4):168–180, Dec 2004.

[1758] S. Yu, D. Cai, J. Wen, and W. Ma. Improving pseudo-relevance feedback in Web information retrieval using web page segmentation. In *Proceedings of the 12th international conference on World Wide Web*, pages 11–18, 2003.

[1759] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *SIGIR 2007: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, July 23-27, 2007*, pages 271–278, 2007.

[1760] B. Yuwono and D. L. Lee. Search and ranking algorithms for locating resources on the World Wide Web. In *Proceedings of the twelfth International Conference on Data Engineering (ICDE)*, pages 164–171, Washington, DC, USA, February 1996. IEEE CS Press.

[1761] B. Yuwono and D. L. Lee. Server ranking for distributed text retrieval systems on the internet. In *Proceedings of the 5th International Conference on Data Systems for Advanced Applications*, 1997.

[1762] R. Zabih, J. Miller, and K. Mai. A feature-based algorithm for detecting and classifying scene breaks. In *MULTIMEDIA '95: Proceedings of the Third ACM International Conference on Multimedia*, pages 189–200, New York, NY, USA, 1995. ACM Press.

[1763] L. Zadeh. Fuzzy sets. In D. Dubois, H. Prade, and R. Yager, editors, *Readings in Fuzzy Sets for Intelligent Systems*. Morgan Kaufmann, 1993.

[1764] O. Zamir and O. Etzioni. Grouper: A Dynamic Clustering Interface to Web Search Results. *Proceedings of the 8th International Conference on World Wide Web (WWW'99)*, 31(11-16):1361–1374, 1999.

[1765] H. Zaragoza, H. Rode, P. Mika, J. Atserias, M. Ciaramita, and G. Attardi. Ranking very many typed entities on Wikipedia. In M. J. Silva, A. H. F. Laender, R. Baeza-Yates, D. L. McGuinness, B. Olstad, Ø. H. Olsen, and A. O. Falcão, editors, *CIKM*, pages 1015–1018. ACM, November 2007.

[1766] D. Zeinalipour-Yazti, V. Kalogeraki, and D. Gunopulos. Information retrieval techniques for peer-to-peer networks. *Computing in Science and Engg.*, 6(4):20–26, 2004.

[1767] D. Zeinalipour-Yazti, V. Kalogeraki, and D. Gunopulos. Exploiting locality for scalable information retrieval in peer-to-peer networks. *Inf. Syst.*, 30(4):277–298, 2005.

[1768] H. J. Zeng, Q. C. He, Z. Chen, W. Y. Ma, and J. Ma. Learning to cluster Web search results. In *Proceedings of the 27th annual international conference on Research and development in information retrieval*, pages 210–217, Sheffield, United Kingdom, 2004. ACM Press.

[1769] Zettair. http://www.seg.rmit.edu.au/zettair/, 2007.

[1770] C. Zhai. *Statistical Language Models for Information Retrieval.* Synthesis Lectures on Human Language Technologies. Morgan Claypool, 2008.

[1771] C. Zhai. Statistical language models for information retrieval: A critical review. *Foundations and Trends in Information Retrieval*, 2(3):137-213, 2009.

[1772] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM TOIS*, 22:179–214, 2004.

[1773] J. Zhang, X. Long, and T. Suel. Performance of compressed inverted list caching in search engines. In J. Huai, R. Chen, H.-W. Hon, Y. Liu, W.-Y. Ma, A. Tomkins, and X. Zhang, editors, *WWW*, pages 387–396, Beijing, China, April 2008. ACM.

[1774] J. Zhang and T. Suel. Efficient Query Evaluation on Large Textual Collections in a Peer-to-Peer Environment. In *P2P'05: Proceedings of the 5th International conference on Peer-to-Peer Computing*, Konstanz, Germany, 2005.

[1775] X. Zhang, F. Junqueira, M. Hiltunen, K. Marzullo, and R. Schlichting. Replicating non-deterministic services on grid environments. In *Proceedings of the IEEE International Symposium on High Performance Distributed Computing (HPDC)*, Haifa, Israel, November 2006.

[1776] Y. Zhang. A Comparison on Open Source Search Engine Software. Technical report, School of Information Sciences and Technology, the Pennsylvania State University, April 2002.

[1777] Z. Zhang and O. Nasraoui. Mining search engine query logs for query recommendation. In *WWW'06: Proceedings of the 15th international conference on World Wide Web*, pages 1039–1040, New York, NY, USA, 2006. ACM.

[1778] Z. Zhang and R. Zhang, editors. *Multimedia Data Mining: A Systematic Introduction to Concepts and Theory.* Data Mining and Knowledge Discovery. Chapman & Hall/CRC, 2008.

[1779] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical report, University of California at Berkeley, Berkeley, CA, USA, 2001.

[1780] Y. Zhao, F. Scholer, and Y. Tsegay. Effective pre-retrieval query performance prediction using similarity and variability evidence. In *Advances in Information Retrieval: 29th European Conference on IR Research*, pages 52–64, 2008.

[1781] E. Zheleva and L. Getoor. To join or not to join: The illusion of privacy in social networks with mixed public and private user profiles. In *WWW 2009*, 2009.

[1782] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *NIPS*, Vancouver, BC, Canada, December 2007. MIT Press.

[1783] B. Zhou and J. H. L. Hansen. Unsupervised audio stream segmentation and clustering via the Bayesian information criterion. In *ICSLP-2000: International Conference on Spoken Language Processing*, pages 714–717, Beijing, China, October 2000.

[1784] X. Zhou and T. Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, 8(6), 2003.

[1785] Y. Zhou and W. B. Croft. Query performance prediction in Web search environments. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 543–550, 2007.

[1786] Q. Zhu, M. A. Gonçalves, R. Shen, L. Cassell, and E. A. Fox. Visual semantic modeling of digital libraries. In *Proc. 7th European Conf. Research and Advanced Technology for Digital Libraries, ECDL*, number 2769 in LNCS, Trondheim, Norway, Aug. 2003. Springer.

[1787] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[1788] Y. Zhu. Enhancing search performance on gnutella-like P2P systems. *IEEE Trans. Parallel Distrib. Syst.*, 17(12):1482–1495, 2006. Senior Member-Hu, Yiming.

[1789] Y. Zhu and Y. Hu. Efficient semantic search on dht overlays. *J. Parallel Distrib. Comput.*, 67(5):604–616, 2007.

[1790] Z. Zhuang, R. Wagle, and C. L. Giles. What's there and what's not?: focused crawling for missing documents in digital libraries. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pages 301–310, Denver, Colorado, 2005.

[1791] S. T. Ziliak and D. N. McCloskey. *The Cult of Statistical Significance: How the Standard Error Costs Us Jobs, Justice, and Lives (Economics, Cognition, and Society)*. Univ. of Michigan, 2008.

[1792] C. Zimmer, C. Tryfonopoulos, and G. Weikum. Exploiting correlated keywords to improve approximate information filtering. In *SIGIR'08: Proceedings of the 31st International ACM SIGIR conference on Research and Development in Information Retrieval*, Singapore, 2008.

[1793] G. Zipf. *Selected Studies of the Principle of Relative Frequency in Language*. Harvard University Press, 1932. Cambridge, MA, USA.

[1794] G. K. Zipf. *Human behavior and the principle of least effort: An introduction to human ecology*. Addison-Wesley, Cambridge, MA, USA, 1949.

[1795] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343, 1977.

[1796] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, 1978.

[1797] J. Zobel. Collection selection via lexicon inspection. In *Proceedings of the Second Australian Document Computing Symposium*, 1997.

[1798] J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Computing Surveys*, 38(2):1–56, 2006.

[1799] J. Zobel, A. Moffat, and K. Ramamohanarao. Inverted files versus signature files for text indexing. *ACM Transactions on Database Systems*, 23(4):453–490, 1998.

[1800] A. Y. Zomaya, editor. *Parallel and Distributed Computing Handbook*. McGraw-Hill, New York, 1996.

索　　引

索引中的页码为英文原书的页码，与书中边栏处标示的页码一致。

R^*-trees（R^*树），392
Ψ function（Ψ 函数），参见 Psi function
5S suite（5S 套件），730

A

A/B testing（A/B 测试），53，169
aboutness（关涉性），574，716
accuracy in text classification（文本分类准确率），326
activation level（激发水平），103
addressing granularity（寻址颗粒度），342
ADI reference collection（ADI 参考集），167
agents（代理），503
agrep（agrep 字符串模糊匹配工具），375，385，395，396
Alexandria library（亚历山大图书馆），2
algebraic models（代数模型），98
algebraic product（代数积），98
Alta Vista（Alta Vista 搜索引擎），459
Amazon（亚马逊），478，485
　mechanical turk（土耳其机器人），171，476
Amdahl's law（阿姆达尔定律），416
AMT（亚马逊土耳其机器人），参见 Amazon mechanical turk
answer quality（答案质量），277
approximate search（近似搜索），385
approximate string matching（近似串匹配），385
approximate-shift-and algorithm（approximate-shift-and 算法），388，391
approximate search（近似搜索）
　word level（词级别），391
architecture of an IR system（信息检索系统架构），5
archives（档案库），719，721
arithmetic coding（算术编码），239，254
ASCII（ASCII 字符集），206
Ask（Ask 引擎），480-484，491，497
association clusters（关联簇），190
association matrix（关联矩阵），191
audio（音频）
　basics（基础知识），602
　browsing（浏览），612
audio and music retrieval（音频和音乐搜索）
　basics（基础知识），602
　fingerprinting（指纹识别），598
　speaker identification（说话人识别），601
　speech recognition（语音识别），599
　　Gaussian mixture models（高斯混合模型），600
　　hidden markov models（隐马尔科夫模型），600
　spoken document retrieval（语音文档检索），602
audio and video（音视频）
　AVSR（音视频语音识别），617
　combining（结合），617
　combining for multimedia（结合音频和视频的多媒体处理），620
audio basics（音频基础知识）
　mel-frequency cepstral coefficients（mel 频率倒谱系数），605
　pitch（音调），603
　sound chromagrams（音谱图），604
　sound spectograms（声谱图），603
　timbre（音色），603
　tone height（音高），605
audio retrieval（音频搜索），597
audio visual speech recognition（音视频语音识别），参见 AVSR
auto-complete（自动填充），28，参见 dynamic query suggestions
autocorrelograms（自相关图），593
automatic global analysis（自动全局分析），195
availability（可用性），425
average precision at *n*（前 *n* 平均精度），参见 precision at *n*（average）
averaged IDF（平均反比文档频率），276
AVSR（音视频语音识别），617

B

bag of words（词袋），62
Baidu（百度），448
Bayes' theorem（贝叶斯定理），122
Bayesian network（贝叶斯网），116
　belief network（信念网），116，122
　　topology（拓扑），122
　belief network ranking（信念网排序），123
　computational cost（计算代价），123
　inference model（推理模型），117

root nodes（根结点），116
benchmark（基准），163
Beowulf cluster（贝奥武夫集群），423
berry picking model（采摘模型），23
biasing（偏置），672
bibliographic databases（文献数据库），698
BibSonomy（BibSonomy 网上文献管理系统），697
BigTable（BigTable 分布式数据库），478
binary independence assumption（二值独立假设），81
binary preferences（二元偏好），参见 BPREF
Bing（Bing 搜索引擎），480-484，488，492，494，495
Binomial model（二项式模型），219
bit-parallelism（位并行），377，379，388
BM1（BM1 模型），105
BM11（BM11 模型），105
BM15（BM15 模型），105
BM25（BM25 模型），104，473
BNDM algorithm（BNDM 算法），381
book（书籍）
contents（内容），13
focus（关注点），12
organization（组织），12
book Web page（本书网站），16
Boolean（布尔）
query（查询），64
Boolean model（布尔模型），60，64
advantages（优点），66
extended（扩展的），92
inference network（推理网），120
visualizing queries（可视化查询），42
Boolean operators（布尔运算符），258
boosting-based ensemble classifiers（增强型集成分类器），318
bot（机器人），参见 crawler
Boyer-Moore's algorithm（Boyer-Moore 算法），374
BPREF（二元偏好），150
BPREF-10（BPREF-10），153
breadcrumbs（面包屑路径），500
browsing（浏览），4，24，498
audio（音频），612
combined with search（结合搜索的），501
directories（目录），499
flat（扁平），499
structure（结构），499
video（视频），612
bucket testing（水桶测试），53，参见 A/B testing
Burrows-Wheeler transform（Burrows-Wheeler 变换），246，254，370
BWT（Burrows-Wheeler 变换），参见 Burrows-Wheeler transform

C

CA Search（化学文摘），699
CACM reference collection（CACM 参考集），167
CAN（内容寻址网络），440
Cassandra（Cassandra 分布式数据库），478
catalogs（编目），716
category（分类）
flat（扁平），35
hierarchical（层次），35
system（系统），35
CCL（通用命令语言），266
CD-RDx（光盘只读数据交换），266
centralized crawler-indexerarchitecture（集中式爬取器-索引器结构），458
chi square（卡方），324
child node（子结点），116
Chord（Chord 协议），440
chromagrams（音谱图），604
CiteseerX（CiteseerX 引文搜索引擎），499
CiteULike（CiteULike 文献管理工具），697
clarity score（清晰度），274
simplified（简化的），277
class of characters（字符类），261
classic IR（经典信息检索），61
classification（分类），参见 text classification
classifier evaluation（分类器评价），291
CLEF collection（CLEF 参考集），167
click（点击）
metric of user preferences（用户偏好指标），187
click graph（点击图），508
clickthrough data（点击数据）
bias（有偏），172
evaluation（评价），171
unbias（无偏），172
clickthrough rate（点击率），504，507
client-server architecture（客户端/服务器架构），404
closed termsets（闭项集），91
cluster-based search（基于聚类的搜索），422
clustering（聚类），35，286
association（关联），190
bisecting K-means（二分 K 均值算法），289
complete link algorithm（完全链算法），198
hierarchical agglomerative（层次式凝聚算法），289
K-means（K 均值算法），286
local（局部聚类），190
local analysis（局部分析），190
Clusty（Clusty 元搜索引擎），508
codes（编码），218

codeword（码字），234
symbol（符号），238
coding tree（码树），238
collection（文档集），参见 reference collection
collection selection（文档集选择），407，433
false drop（误检），407
combination of evidential sources（结合证据源），121
communication（通信），425
comparing search software（搜索软件比较），743
comparison of classic IR models（经典信息检索模型比较），86
complete link algorithm（完全链算法），198
compression（压缩），参见 text compression
inverted index（倒排索引），354
preprocessing（预处理），234
sequential search（序列搜索），389
software（软件），206
suffix array（后缀数组），367
computational advertising（计算广告学），504
concept（概念），194
concept space（概念空间），101，122
conjunctive component（合取分量），64
content-based image retrieval（基于内容的图像检索），592
context（上下文，语境，情境）
text（文本），257
user（用户），267
contextualization（情境化），675
controlled vocabulary（受控词汇表），229
CORI（CORI 文档集选择算法），406，408
correlation（相关性）
index terms（索引项），68
matrix（矩阵），96
term-term（项间），67
correlograms（相关图），593
cosine（余弦）
similarity formula（相似度公式），78
cosine distance（余弦距离），223
coverage ratio（覆盖率），144
Cranfield paradigm（Cranfield 范式），132
crawler（爬虫），515
architecture（架构），521
bandwidth usage（带宽使用），538
coverage（覆盖度），517
distributed（分布式），543
DNS resolution（DNS 解析），523
duplicated pages（重复网页），525
evaluation（评价），539
focused（聚焦），518，530
format specific（特殊格式），518
freshness（新鲜度），519，531
characterization（刻画），534
estimating（估计），533
strategies（策略），532
help of web sites（在网站帮助下），542
hidden web（暗网），541
identification（识别），536
mirroring（镜像），518
modular（模块化），544
news（新闻），517
open-source（开源），544
parallel（并行），526
parsing（解析），524
quality（质量），517，519
revisit policy（重访问策略），530
robot exclusion（机器人排除），537
scheduler（调度器），521
scheduling algorithms（调度算法），527
seed pages（种子网页），516
selection policy（选择策略），528
soft-404 pages（软 404 页面），525
storage（存储），522
topical（主题），518
URL canonization（URL 规范化），524
URL frontier（URL 前沿），522
vertical（垂直），517
volume（容量），520
cross-validation in text classification（文本分类中的交叉检验），329
crowdsourcing（众包），170
CSS（层叠样式表），211
CTR（点击率），参见 clickthrough rate
Cuil（Cuil 搜索引擎），488
curse of dimensionality（维度灾难），394
Cyrillic（西里尔字母），448
Cystic Fibrosis reference collection（囊肿性纤维化参考集），167

D

DAG（有向无环图），116
DARPA（国防部高级研究计划局），159
data retrieval（数据检索），5，255
data，semi-structured（半结构化数据），503
DCG（折扣累积增益），145
basic（基本），146
curves（曲线），147
ideal（理想的），148
normalized（归一化），149
decision trees（决策树），294
deep links（深度链接），32

degree of membership（隶属度），95
degree of similarity（相似度），77
Delicious（Delicious 社区化网页书签服务网站），232
dense coding（密集编码），239，242，254
　(s，c) -dense codes（(S，C) 密集码），244
dependability（可靠性），425
DFA（有限自动机），参见 finite automaton
DFR（随机差异模型），113
DHT（分布式散列表），参见 distributed hash table
dictionary（词典），参见 vocabulary
dictionary methods（词典压缩方法），245
digital libraries（数字图书馆），3，711
　5S（5S 套件），730
　architecture（架构），713
　archives（档案库），719
　archiving（归档），721
　collections（馆藏），716
　digitization（数字化），715
　Dublin Core（都柏林核心），716
　economical issues（经济问题），726
　ETANA-DL（ETANA-DL 考古数字图书馆），732
　handles（句柄），714
　MARC（机器可读目录），716
　metadata and catalogs（元数据和目录），716
　METS（元数据编码和传输标准），717
　MODS（元数据对象描述框架），717
　NDLTD（联网学位论文数字图书馆），731
　NSDL（国家科学数字图书馆），732
　objects and collections（对象和馆藏），714
　ODL（开放数字图书馆），729
　preservation（保存），721
　repositories（资源库），719
　services（服务），723
　smart objects（智能对象），716
　social issues（社会问题），725
　software systems（软件系统），727
digital search tree（数字搜索树），参见 trie
digitization（数字化），715
dimensionality reduction（降维），320
directed acyclic graphs（有向无环图），参见 DAG
discounted cumulated gain（折扣累积增益），参见 DCG
disjunctive normal form（析取范式），64
display of retrieval results（检索结果显示），29
distance function（距离函数），222
distortion-based views（变形视图），42
distributed cache table（分布式缓存表），443
distributed computing（分布式计算），415，424
distributed hash table（分布式散列表），440
distributed search（分布式搜索），400，403，424
　cluster-based（基于聚类的），400，403，422
　collecting data（数据收集），426
　collection partitioning（文档集划分），405
　communication（通信），402，427
　dependability（可靠性），427
　external factors（外部因素），425
　indexing（索引），426，428
　　communication（通信），430
　　dependability（可靠性），429
　　external factors（外部因素），430
　query processing（查询处理），400，426，431
　　communication（通信），436
　　components（组件），431
　　connectivity（可连接性），432
　　dependability（可靠性），435
　　external factors（外部因素），437
　　interaction（交互），432
　　load balancing（负载均衡），433
　　roles（角色），432
　server software（服务器软件），402
　Web issues（Web 问题），437
divergence from randomness（随机差异模型），参见 DFR
document（文档），203
　concepts（概念），194
　format（格式），204，206
　length normalization（长度归一化），75
　logical view（逻辑视图），63
　metadata（元数据），205
　non-relevant（不相关），65
　organization（组织），231
　preprocessing（预处理），223
　presentation（展示），204
　relevant（相关），65
　representation（表示），58，62
　semantics（语义），203，204
　semi-static collection（半静态文档集），338
　similarity（相似度），223
　structure（结构），64
　style（样式），204
　syntax（句法），203
　vocabulary（词汇表），221
　word distribution（词汇分布），220
document collection（文档集）
　word distribution（词汇分布），221
document partitioning（文档划分），405
doubling search（成倍扩展搜索），346
DTD（文档类型定义），209
DTF（决策理论框架），408
Dublin Core（都柏林核心），716

dynamic（动态）
query suggestions（查询建议），28，484
dynamic programming（动态规划），385
dynamic ranking（动态排序），674
dynamic search（动态搜索），503
dynamic summaries（动态文摘），609

E

E-measure（E 测度，E 值），143
e-publishing era（电子出版时代），9
edit distance（编辑距离），222，261，385
editorial judgements（编辑判断），474
effectiveness（有效性），337
efficiency（效率），337
elementary concept（基本概念），122
Elias-δ code（Elias-δ 编码），355
Elias-γ code（Elias-γ 编码），355
employee dissatisfaction in search（员工搜索不满意度），667
Endeca（Endeca 企业搜索引擎），495
ensemble classifiers（集成分类器），316
boosting-based（增强型），318
stacking-based（叠加型），317
enterprise search（企业搜索），126，641
architecture（架构），648
context and personalization（情境化和个性化），668
contextualization（情境化），675
employee dissatisfaction（员工不满意度），667
evaluation（评价），662
TREC（文本检索会议），663
indexing（索引），652
metasearch（元搜索），659
personalization（个性化），668
biasing（偏置），672
dynamic ranking（动态排序），674
information filtering（信息过滤），679
scoping（范围），672
social recommender systems（社会化推荐系统），680
user modeling（用户建模），677
user profiles（用户轮廓），677
query processing（查询处理），654
results presentation（结果展示），655
security models（安全模型），657
software（软件），643
tasks（任务），644
textual annotations（文本注释），653
tuning（调试），665
workplace（工作场所），644
entropy（熵），219，234，238，239
epistemological view（认识论观点），117
error in text classification（文本分类错误），326
ETANA-DL（ETANA-DL 考古数字图书馆），732
evaluation（评价），参见 retrieval evaluation
enterprise search（企业搜索），662
open source software（开源软件），742
R-precision（R 精度），141
exact string matching（严格串匹配），372
exhaustively（穷尽性），70
expected search length in evaluation（期望搜索长度），176
experimentation in the lab（实验室中的实验），168
explicit feedback（显式反馈），177，178，180
exploratory search（探索式搜索），22
extended Boolean model（扩展布尔检索模型），92
eye tracking（眼动追踪），185

F

F-measure（F 测度，F 值），139，144
text classification（文本分类），327
faceted（分面）
categories（分类），35
metadata（元数据），36
false drop（误检），358，407
feature（特征）
selection（选择），292，320
vectors（向量），292
federated search（联合搜索），399，404，438
collection selection（文档集选择），407
local（局部），403
query processing（查询处理），439
federated system（联合系统），432
feedback（反馈）
evaluation（评价），184
explicit（显式），177，178，180
implicit（隐式），177，179，195
relevance（相关性），参见 relevance feedback
filtering（过滤），164
filtration（筛选），388
fingerprinting（音频指纹识别），598
finite automaton（有限自动机）
deterministic（确定性），377，388
non-deterministic（非确定性），375，386
over codewords（码字上的），390
suffix based（基于后缀的），380
flat browsing（扁平浏览），499
flat categories（扁平分类），35
flat indexing lists（扁平索引列表），549
focused crawling（聚焦爬取），518
folksonomy（分众分类法），232
frequentist view（频率论观点），117
freshness（新鲜度），531

full evaluation（完全评价），348
full inverted index（完全倒排表），341
fusion model（融合模型），614
  naming faces（人脸命名），614
fusion models（融合模型）
  audio and video（音视频），617
  AVSR（音视频语音识别），617
  naming audio（音频命名），616
  naming images（图像命名），615
fuzzy（模糊）
  Boolean（布尔），259
  set model（模糊集模型），95
  set theory（模糊集理论），95
  thesaurus（同义词典），96
fuzzy IR（模糊信息检索），96
  correlation matrix（相关性矩阵），96
  document sets（文档集），97
  ranking（排序），98
fuzzy set（模糊集）
  algebraic product（代数和），98
  algebraic sum（代数和），96
  complement（补集），95
  intersection（交集），95
  max（极大），96
  min（极小），96
  union（并集），95

G

Gaussian mixture models（高斯混合模型），600
generalized term vector（广义索引项向量），100
generalized vector model（广义向量模型），98
GFS（谷歌文件系统），参见 Google File System
global analysis（全局分析），179，194，195
Gnutella（Gnutella 文件共享系统），440
golden set（基准集），参见 ground truth
Golomb code（Golomb 码），356
Google（谷歌），471，480-484，487，490，491，494-496
  ads（广告），488
  blog（博客），490，491
  cache（网页快照），496
  did you mean（您是不是要找），493
  I feel lucky（手气不错），482
  image search（图像搜索），480，488
  notebook（记事本），497
  numrange（数值范围），482
  similar（类似结果），496
  suggest（建议），484-487
  translate（翻译），496
  universal search（统一搜索），491
Google File System（谷歌文件系统），478
GOV collection（GOV 文档集），162
GPX（GPX 算法），559
granularity（颗粒度），425
grep（全面正则表达式搜索软件），337，395
grid computing（网格计算），423
grid-files（网格文件），392
ground truth（真实情况），475

H

Hadoop，424，478
  HDFS（Hadoop 分布式文件系统），478
Hamming（海明），222
handles（句柄），714
harmonic mean（调和平均），144
HDFS（Hadoop 分布式文件系统），478
Heaps' Law（Heaps 法则），221
Heaps' law（Heaps 法则），237，343，344，455
heavy tail（重尾），455
hierarchical agglomerative clustering（层次式凝聚聚类），289
hierarchical index（层次索引），550
hierarchical organization（层次组织），35
highlighting（加亮显示），31
HITS（超文本推导主题搜索算法），470
Horspool's algorithm（Horspool 算法），373
hop-plot（hop 图），457
how people search（人们如何搜索），21
HTML（超文本标记语言），124，211
HTTP（超文本传输协议），124
  user-agent（用户代理），537
hub（枢纽），457
hubs & authorities（枢纽页和权威页），125
Huffman coding（霍夫曼编码），238，239，254
  byte（字节），240
  canonical（规范），240，254
human experimentation in the lab（实验室中的人工实验），168
hyperspace（超空间），125
hypertext（超文本），124，501
  transfer protocol（传输协议），参见 HTTP
  link（链接），124
  markup language（标记语言），参见 HTML
  navigation（导航），124
  Web（万维网），125
HyTime（超媒体/基于时间的结构语言），217

I

IDF（反比文档频率），70
ILS（图书馆集成系统），687

image（图像）
format（格式），207
image retrieval（图像检索）
color-based（基于颜色的），593
content-based（基于内容的），592
salient points（显著点），596
texture-based（基于纹理的），593
implicit feedback（隐式反馈），177，179
independence of index terms（索引项独立性），98
index（索引），7，338
inverted（倒排），340
partial（部分），353
signature file（签名文件），357
space（空间），338
storage（存储），338
suffix array（后缀数组），360
suffix tree（后缀树），360
index partitioning（索引划分），400
term-based（基于项的），412
index term（索引项），57，61-63，227，229
Boolean model（布尔模型），64
correlation（相关性），68，96
probabilistic weight（概率权重），80
vector model（向量模型），77
vector weight（向量权重），77
vector weighting（向量赋权），72
weights（权重），66
indexing（索引）
block addressing（块寻址），343
document addressing（文档寻址），343
enterprise search（企业搜索），652
file-addressing（文件寻址），343
incremental（增量），749
index size（索引大小），747
inverted index（倒排索引），351
multidimensional（多维），391
signature file（签名文件），359
space（空间），338
suffix array（后缀数组），366
suffix tree（后缀树），365
time（时间），338，745
word-addressing（词寻址），343
WT10g collection（WT10g 文档集），748
INEX（INEX XML 检索评测），566
collection（文档集），166
inference network（推理网），116
Boolean model（布尔模型），120
combining evidences（结合证据源），121
priors（先验概率），119
ranking（排序），119
simplifying assumption（简化假设），118
TF-IDF ranking（TF-IDF 排序），121
topology（拓扑），117
inference network model（推理网模型），117
information（信息）
gain（增益），323
foraging theory（搜寻理论），24
lookup（查找），22
request（需求），162，163
scent（线索），24
information filtering（信息过滤），679
information need（信息需求），3
information retrieval（信息检索），参见 *IR*
information seeking（信息搜寻）
classic model（经典模型），23
dynamic model（动态模型），23
information theory（信息论），218
informativeness（信息性），139
InkML（墨水标记语言），*216*
Inquery（Inquery），408
instantaneous code（即时码），238
integrated library system（图书馆集成系统），参见 library，ILS
interactive video summaries（交互式视频摘要），611
interlaced search（交错搜索），381
Internet Archive（Internet 档案馆），519，543
interpolated precision（插值精度），137
intranet（企业网），447
inverse document frequency（反比文档频率），参见 IDF
inverse term frequency（反比项频），195
inverted file（倒排文件），参见 inverted index
inverted index（倒排索引），7，340
character position（字符位置），341
compressed（压缩），354
construction（构建），351
external（外部），353
incremental（增量），354
intermittent（间歇），354
internal（内部），351
merging indexes（索引合并），353
rebuild（重建），354
full（完全），341
in SIMD architecture（单指令流多数据流架构），419
inverted list（倒排表），340
list intersection（列表求交集），346
partitioning（划分）
term-based（基于项的），412
ranking（排序），348
search（搜索），345

Boolean queries（布尔查询），348
complex queries（复杂查询），346
multiple words（多个词），345
phrase（短语），347
proximity（邻近），347
single word（单个词），345
structural queries（结构化查询），357
structure（结构），340
word position（词位置），341
inverted list（倒排表），340
compressed（压缩），354
IR（信息检索）
classic（经典），61
computer centered view（以计算机为中心的角度），1
early developments（早期发展），1
fuzzy（模糊），96
human centered view（以人为中心的角度），1
library（图书馆），3
models（模型），57
taxonomy（分类体系），59
multimedia（多媒体），588
problem（问题），4
system architecture（系统架构），5
ISI reference collection（ISI 参考集），167

J

Jelinek-Mercer method（Jelinek-Mercer 方法），110

K

K-means（*K* 均值）
bisecting（二分），289
clustering（聚类），286
k-NN classifier（*k* 近邻分类），299
Kanji（汉字），448，450
Kendall Tau coefficient（肯德尔等级相关系数），156
keyword（关键字，关键词），57，61，227
connection matrix（连接矩阵），96
in context（上下文内），31
thesaurus（同义词典），229
Knuth-Morris-Pratt's algorithm（KMP 算法），374
KWIC（上下文关键字），31

L

language model（语言模型），107
Bernoulli process（伯努利过程），111
multinomial process（多项过程），108
smoothing（平滑），109
statistical foundation（统计基础），107
languages（语言）
markup（标记），208
latent semantic indexing（潜在语义索引），参见 LSI
lazy evaluation（惰性求值），348
Levenshtein distance（Levenshtein 距离），222，261，385
lexical analysis（词汇分析），224
lexicon（词典），参见 vocabulary
LFU（近期最少使用算法），463
library（图书馆），3，685
ILS（图书馆集成系统），687，691
products（产品），695
vendors（供应商），695
information environment（信息环境），685
integrated system（集成系统），687
OPAC（联机公共检索目录），687
linear quadtrees（线性四叉树），392
link analysis（链接分析），456
macroscopic（宏观），456
mesoscopic（介观），457
microscopic（微观），457
LISA reference collection（LISA 参考集），167
list intersection（列表求交集），346
load balancer（负载均衡器），423，460
load balancing（负载均衡），423
local analysis（局部分析），179，190，194
local association matrix（局部关联矩阵），191
local clustering（局部聚类），190
local context analysis（局部上下文分析），193，194
local scalar matrix（局部标量矩阵），192
Log Normal distribution（对数正态分布），455
logical view of documents（文档的逻辑视图），63
longest common subsequence（最长公共子序列），223
longitudinal study（纵向研究），53
lost in hyperspace（迷失在超空间），125
LRU（最近最少使用算法），463
LSI（潜在语义索引），101
Luhn assumption（Luhn 假设），68
LZCS（LZCS 结构化文本压缩方法），250，254

M

machine learning（机器学习），282
Mandelbrot distribution（Mandelbrot 分布），220
MAP（平均精度均值），139，140
MARC（机器可读目录），205，266，689，716
XML（可扩展标记语言），215
markup（标记），208
markup languages（标记语言），208
HTML（超文本标记语言），211
SGML（标准通用标记语言），209

match point（匹配点），549
MathML（数学标记语言），215
max fuzzy（模糊极大），96
mean average precision（平均精度均值），参见 MAP
mean reciprocal rank（平均排序倒数），参见 MRR
measure（测度）
  F（F 测度，F 值），144
MED reference collection（MED 参考集），167
mel-frequency cepstral coefficients（mel 频率倒谱系数），605
merging（合并），345，353
metadata（元数据），205
  descriptive（描述性），205
  Dublin Core（都柏林核心），716
  graphics（图形），208
  MARC（机器可读目录），716
  MODS（元数据对象描述框架），717
  semantics（语义），205
metadata and catalogs（元数据和目录），716
metasearch（元搜索），400，508
  enterprise（企业），659
metric access methods（度量访问方法），392
metric cluster（度量簇），191
metrics for text classification（文本分类评价指标），325
METS（元数据编码和传输标准），717
MIMD（多指令流多数据流），415
MIME（多用途互联网邮件扩展格式），206
min fuzzy（模糊极小），96
minterm（最小项），99
mirroring（镜像），518
MISD（多指令流单数据流），415
ML（机器学习），参见 machine learning
model（模型）
  algebraic（代数），60，98
  Bayesian network（贝叶斯网），116
  belief network（信念网），122
  Boolean（布尔），64
  comparison（比较），86
  extended Boolean（扩展布尔），92
  fuzzy set（模糊集），95
  generalized vector（广义向量），98
  hypertext（超文本），124
  inference network（推理网），117
  language（语言），107
  LSI（潜在语义索引），101
  neural network（神经网），102
  others（其他），124
  probabilistic（概率），60，79，104
  set theoretic（集合论），60，87
  set-based（基于集合的），87
  vector（向量），77
  Web-based（基于 Web 的），125
modeling（建模），57
  definition（定义），58
  taxonomy（分类体系），59
mosaics（图像拼接），608
MPEG（MPEG 标准），625
  color（颜色），626
  intensity and sampling（强度和采样），626
  lossy compression（有损压缩），628
  lossyless compression（无损压缩），628
  motion prediction（运动预测），630
  MPEG-1（MPEG-1 标准），633
  MPEG-2（MPEG-2 标准），633
  MPEG-21（MPEG-21 标准），635
  MPEG-4（MPEG-4 标准），635
  MPEG-7（MPEG-7 标准），635
  standards（标准），633
  temporal redundancy（时间冗余），630
MPI（消息传递接口），424
MRR（平均排序倒数），139，142
multimedia（多媒体），207，587
  audio and video（音视频），620
  compression（压缩），625
  data segmentation（数据分割），620
  IR（信息检索），参见 multimedia IR
  machine-generated data（机器生成的数据），591
  MPEG（MPEG 标准），625
  segmentation（分割），620
  speech segmentation（语音分割），624
  video segmentation（视频分割），620
multimedia IR（多媒体信息检索），126，587，588
  audio and music retrieval（音频和音乐检索），597
  content-based image retrieval（基于内容的图像检索），592
  feature ambiguity（特征歧义性），591
  fusion model（融合模型），614
  semantic gap（语义鸿沟），589
  versus text IR（与文本信息检索的对比），589
  video retrieval（视频检索），606
multitasking（多任务），416
music retrieval（音乐检索），597
mutual information（互信息），323

N

n-termset，（*n* 项集），88
naive Bayes classifier（朴素贝叶斯分类器），303
  binary independence（二值独立），303
  multinomial（多项），305
naming audio（音频命名），616

naming faces（人脸命名），614
naming images（图像命名），615
Napster（Napster 联机音乐商店），440
National Science digital library（国家科学数字图书馆），732
navigation（导航），4，24
  versus search（与搜索），24
NDLTD（联网论文数字图书馆），731
neighbor terms（相邻项），193
network topology（网络拓扑），117
neural network（神经网），102
  model（模型），102
  TF-IDF ranking（TF-IDF 排序），104
  thesaurus（同义词典），103
Newsgroups collection（Newsgroups 文档集），166
NEXI（NEXI 查询语言），578
NFA（非确定有限自动机），参见 finite automaton
NLM reference collection（NLM 参考集），167
node of text（文本结点），549
non independence of index terms（索引项非独立性），98
non-binary weights（非二元权重），77
non-cooperative system（非合作系统），433
non-overlapping lists（非覆盖列表），549
non-relevant document（不相关文档），65
normalized TF-IDF（归一化 TF-IDF），93
noun groups（名词组），62，63，194
nouns（名词），62
novelty ratio（新颖率），145
NPL reference collection（NPL 参考集），167
NSDL（国家科学数字图书馆），732
NTCIR collection（NTCIR 文档集），166
NY Times（纽约时报数据库），699

O

objects and collections（对象和馆藏），714
occurrence list（记录表），340
occurrences（记录），340
OCLC（联机计算机图书馆中心），689，703
odds of relevance（相关度优势比），80
ODL（开放数字图书馆），729
ODP（开放目录计划），452，499
OHSUMED collection（OHSUMED 文档集），166
Online Public Access Catalogue（联机公共检索目录），参见 OPAC
OPAC（联机公共检索目录），687
  bibliographic records（书目记录），689
  integration（集成），693
  interface（界面），691
  users（用户），694
Open Linking Data（开放链接数据项目），510
open source search engines（开源搜索引擎），737
  global evaluation（全局评价），752
open system（开放系统），433
orienteering（定向），23
OWL（互联网本体语言），217

P

p-norms（p 范式），93
P2P（对等），401，404
  retrieval（检索），440
panning（平移），42
parallel
  flights（平行航班测试），53
parallel computing（并行计算），414
parallel search（并行搜索），401，414
  collection partitioning（文档集划分），405
  communication（通信），402
  efficiency（效率），416
  inverted index（倒排索引）
    in SIMD architecture（单指令流多数据流架构），419
  MIMD（多指令流多数据流），403，416
  number of processors（处理器数量），402
  SIMD（单指令流多数据流），403，418
  speedup（加速比），416
parents of a node（父结点），116
Pareto distribution（帕累托分布），456
partitioning（划分），425
  document collection（文档集），404，405
  index（索引），404
  inverted index（倒排索引），409
    logical document（逻辑文档），409
    physical document（物理文档），410
  signature files（签名文件），413
  suffix arrays（后缀数组），413
passage（段落），194，203
PAT array（PAT 数组），参见 suffix array
PAT tree（PAT 树），参见 suffix tree
pattern（模式），260
  extended（扩展），261
  prefix（前缀），260
  regular expression（正则表达式），261
  substring（子串），260
  suffix（后缀），260
  with errors（容错），260
  word（词），260
pattern matching（模式匹配），260，参见 sequential search
pattern，range（模式，范围），260
PDC（概率驱动缓存），463
PDF（便携文档格式），206

peer-to-peer architecture（对等架构），参见 P2P
Persin's algorithm（Persin 算法），349
personalization（个性化）
  biasing（偏置），672
  dynamic ranking（动态排序），674
  enterprise（企业），668
  information filtering（信息过滤），679
  scoping（范围），672
  social recommender systems（社会化推荐系统），680
  user modeling（用户建模），677
  user profiles（用户轮廓），677
phrase（短语）（in text compression（文本压缩）），234
phrase browsing（短语浏览），246
pitch（音调），603
pooling method（聚合方法），163
Porter's algorithm（Porter 算法），227
posting file（记录文件），参见 inverted list
power law（幂率），454
PPM（部分匹配预测），236，254
precision（准确率，查准率，精度），135
  appropriateness（合理性），139
  at *n*（前 *n*），139
  at *n*（average）（前 *n* 平均），140
  average（平均），138
  defined（定义），135
  disadvantages（缺点），139
  histogram（直方图），141
  interpolated（插值），137
  text classification（文本分类），327
preferential attachment（优先连接），457
prefix（-free）code（前缀（无关）码），238
preservation（保存），721
prior probability（先验概率），119
  distribution in inference networks（推理网分布），119
privacy of queries（查询隐私），512
probabilistic model（概率模型），60，79
  advantages（优点），86
  alternative（其他），104
  disadvantages（缺点），86
  fundamental assumption（基础假设），80
  ideal answer set（理想答案集），80
  ranking formula（排序公式），82
  ranking principle（排序原则），80
  term reweighting（项再赋权），183
  term weight（项权重），80
probability（概率）
  epistemological view（认识论观点），117
  frequentist view（频率论观点），117
  prior（先验），119
proximal nodes（相邻结点），550
pseudo-document（伪文档），102
Psi function（Ψ 函数），368
PVM（并行虚拟机），424

Q

query（查询），4
  allowing errors（容错），260
  ambiguity（歧义），273
  auto-complete（自动填充），28
  automatic construction（自动构建），165
  average length（平均长度），267
  basic（基本），256
  Boolean（布尔），64，258
  Boolean operations（布尔操作），258
  chain（链），188，274
  characteristics（特征），267
  conjunctive component（合取分量），62，64
  context（上下文），257
  correlation with text（文本相关性），268
  difficulty（难度），274
    clarity score（清晰度），274
    post-retrieval algorithms（检索后算法），274
    pre-retrieval algorithms（检索前算法），276
  distribution（分布），268
  extended pattern（扩展模式），261
  fuzzy Boolean（模糊布尔），259
  heavy tail（重尾），273
  informational（信息型），270
  intent（意图），270
  intent prediction（意图预测），271
  interfaces（界面），27
  keyword（关键词），256
  languages（语言），255
  latency（时延），338
  log（日志），267
  logical session（逻辑会话），274
  long tail（长尾），268
  looking at results（结果查看），268
  manual construction（人工构建），165
  mission（任务），273
  natural language（自然语言），262
  navigational（导航型），270
  number of clicks（点击数），269
  pattern matching（模式匹配），260
  phrase（短语），257
  properties（属性），267
  protocols（协议），265
  proximity（邻近），258

range（范围），260
refinding（重新），271
reformulation（重构），32，274
regular expression（正则表达式），261
representation（表示），58，62
session（会话），273
single-word（单个词），256
specification（描述），26
structural（结构化），262
suggestions（建议），28
syntax tree（句法树），258，348
term distribution（项分布），268
term highlighting（项加亮显示），31
throughput（吞吐量），338
time dynamics（动态性），269
topic（主题），272
transactional（事务型），270
query expansion（查询扩展），32，177，181，192
based on a similarity thesaurus（基于相似度同义词典），195
neighbor terms（邻居项），192
statistical thesaurus（统计同义词典），198
through local context analysis（通过局部上下文分析），193
query feedback（查询反馈），276
query mining（查询挖掘），507
query processing（查询处理）
enterprise search（企业搜索），654
query routing（查询路由），407
query scope（查询范围算法），277
quick links（快速链接），501

R

R-precision（R 精度），141
RAID（独立冗余磁盘阵列），417
ranking（排序），7，57，58
belief network vector（信念网向量），123
Boolean inference network（布尔信念网），120
correlation metrics（相关性测度），153
enterprise（企业），674
function（函数），58
fuzzy model（模糊模型），98
inference network（推理网），119
Kendall Tau coefficient（肯德尔等级相关系数），156
neural network model（神经网模型），104
probabilistic model（概率模型），82
process（过程），7
Spearman coefficient（斯皮尔曼系数），154
TF-IDF for inference network（推理网 TF-IDF），121
vector model（向量模型），78
Web search（Web 搜索），468
RDF（资源描述框架），205，216
schema（模式），217
recall（召回率，查准率），135
appropriateness（合理性），139
defined（定义），135
disadvantages（优点），139
effort（代价度），145
relative（相对），145
standard levels（标准水平），136
text classification（文本分类），327
reduced concept space（约化概念空间），101
reference collection（参考集），134，158
region of text（文本区域），549
regular expressions（正则表达式），261，382
operations（操作），261
related term suggestions（相关项建议），32
relevance（相关性），4
judgements（判断），185
relevance feedback（相关反馈），177，180
evaluation（评价），184
probabilistic model（概率模型），183
through clicks（通过点击），185
relevant document（相关文档），65
reliability（可依赖性），425
replication（复制），400
repositories（资源库），参见 archives
resemblance（类似度），223
residual collection（剩余文档集），184
Resource Description Framework（资源描述框架），参见 RDF
retrieval（检索）
percentage（比例），749
data（数据），5
information（信息），5
process（过程），7
results display（结果显示），29
unit（单元），256
retrieval evaluation（检索评价），131
BPREF（二元偏好），150
DCG（折扣累积增益），145
E-measure（E 值），143
F-measure（F 值），144
frustration（挫折度），176
MAP（平均精度均值），140
MRR（平均排序倒数），142
practical caveats（实践说明），173
precision（精度，查准率），135
rank correlation metrics（排序相关性测度），153
recall（召回率，查全率），135
relevance feedback（相关反馈），184

satisfaction（满意度），176
search interfaces（搜索界面），50
user oriented（面向用户），144
using clickthrough data（使用点击数据），171
Reuters collection（Reuters 文档集），166
rich-get-richer effect（富者益富效应），457
Rocchio（Rocchio 方法）
relevance feedback（相关反馈），180
Rocchio classifier（Rocchio 分类器），300
root nodes（根结点），*116*
routing（分流），164
RTF（富文本格式），206

S

safety（安全性），425
salient points（显著点），596
salient stills（跳跃剧照），608
scalar cluster（标量簇），192
scalar matrix（标量矩阵），192
scalar similarity measure（标量相似度测度），192
scale free networks（无尺度网络），457
SCM（结构化上下文建模压缩方法），250，254
scoping（范围），672
SDC（静态动态相结合的缓存策略），463，464
search（搜索）
approximate（近似），385
benchmark example（基准测试用例），743
classes of characters（字符类），378
compressed text（压缩文本），*389*
distributed（分布式），399，424
enterprise（企业），641，706
exploratory（探索式），22
interfaces（界面），25
interfaces evaluation（界面评价），50
inverted index（倒排索引），345
multidimensional（多维），391
multiple patterns（多重模式），384
open source（开源），737
organizations（机构），706
parallel（并行），*399*，414
quality comparison（质量比较），751
regular expressions（正则表达式），382
results（结果），30，35
results presentation（结果展示），655
results visualization（结果可视化），40
sequential（序列），372
signature file（签名文件），359
single word（单个词），345
structured text（结构化文本），545
suffix tree（后缀树），362
time（时间），749
versus navigation（与导航），24
Web（万维网），447
wild cards（通配符），379
XML（可扩展标记语言），545
search engine（搜索引擎），参见 Web search
results page（结果页面），30
searchonym（搜索同义项），193
security（保险性），425
enterprise search（企业搜索），657
segmentation（分割），620
self-indexes（自索引），367
self-synchronizing codes（自同步码），243
semantic indexing（语义索引），101
semantic Web（语义网），450
semi-static collection（半静态文档集），338
semi-supervised learning（半监督学习），*283*
sensemaking（意义建构），22
SEO（搜索引擎优化），参见 Web search optimization
sequencing（序列），124
sequential search（序列搜索），372
approximate（近似），385
automata（自动机），375
bit-parallelism（位并行），377
BNDM algorithm（BNDM 算法），381
Boyer-Moore（Boyer-Moore 算法），374
brute force（暴力），372
classes of characters（字符类），378
complex patterns（复杂模式），375
compressed text（压缩文本），389
dynamic programming（动态规划），385
filtration（筛选），388
Horspool（Horspool 算法），373
Knuth-Morris-Pratt（KMP 算法），374
multiple patterns（多重模式），384
regular expressions（正则表达式），382
shift-and algorithm（shift-and 算法），377
simple string（简单字符串），373
suffix automata（后缀自动机），380
Sunday（Sunday 算法），374
Thompson（Thompson 构建过程），382
wild cards（通配符），379
SERP（搜索引擎结果页面），30，参见 Web search，result page
set intersection（列表求交集），346
set theoretic models（集合论模型），87
set-based model（基于集合的模型），87
SFQL（结构化全文查询语言），*266*
SGML（标准通用标记语言），161，204，209

SHA（安全散列算法），479
shift-and algorithm（shift-and 算法），377
shift-and-extended algorithm（扩展 shift-and 算法），379，390
shift-or algorithm（shift-or 算法），396
shingles（片段），480
shopbot（网店机器人），参见 crawler
side-by-side panels（并排面板），168
signature（签名），358
signature file（签名文件），357，359
  compressed（压缩），359
  construction（构建），359
  search（搜索），359
  structure（结构），358
SIMD（单指令流多数据流），415
similarity thesaurus（相似度同义词典），195
SISD（单指令流单数据流），415
sitelinks（站内链接），32
sitemaps（网站地图），500
SMIL（同步多媒体集成语言），218
smoothing（平滑）
  Jelinek-Mercer method（Jelinek-Mercer 方法），110，556
snippet（片段），341，489
snippets（片段），30，476
social networks（社交网络），513
social recommender systems（社会化推荐系统），680
software architecture（软件架构），5
sound chromagrams（音谱图），604
sound spectograms（声谱图），603
Soundex（Soundex 语音算法），261
source selection（源选择），407
spaceless words model（无空格单词模型），237
spam（垃圾），参见 Web spam
spambot（垃圾机器人），参见 crawler
spamdexing（垃圾索引），参见 Web spam
spatial access methods（空间访问方法），391
speaker identification（说话人识别），601
Spearman coefficient（斯皮尔曼系数），154
specificity（特异性），70
spectograms（谱图），603
speech recognition（语音识别），599
  Gaussian mixture models（高斯混合模型），600
  hidden markov models（隐马尔科夫模型），600
speech segmentation（语音分割），624
speedup（加速），416
spider（蜘蛛），参见 crawler
spoken document retrieval（语音文档检索），602
sponsored results（广告搜索结果），488
sponsored search（广告搜索），504
spread activation（激活扩散），102
stacking-based ensemble classifiers（叠加型集成分类器），317
standard collections for textclassification（文本分类标准文档集），329
starfields displays（星空显示），47
STARTS（斯坦福互联网元搜索协议），438
statistical text model（统计文本模型），218
statistical thesaurus（统计同义词典），198
stemming（词干提取），63，226
  Porter's algorithm（Porter 算法），227
STEP（产品模型数据交换标准），251
Stirling's formula（Stirling 公式），115
stopwords（禁用词），63，220
  elimination（去除），226
storyboards（故事板），607
structured search（结构化搜索），545
structured text（结构化文本），357
structured text retrieval（结构化文本检索），126，参见 XML retrieval
suffix（后缀），360
suffix array（后缀数组），361
  compressed（压缩），367
  construction（构建），366
suffix automata（后缀自动机），380
suffix tree（后缀树），361
  construction（构建），365
  search（搜索）
    complex patterns（复杂模式），363
    string（串），362
suffix trie（后缀 trie 树），361
suggestions（建议）
  related terms（相关项），32
summary table statistics（摘要统计表），144
Sunday's algorithm（Sunday 算法），374
supervised algorithms（监督算法），291
  decision trees（决策树），294
supervised learning（监督学习），283
support vector machines（支持向量机），参见 SVM
SVG（可缩放矢量图形），216
SVM（支持向量机），307，474
SVM classifier（SVM 分类器），306
symbol code（符号码），238
synonym（同义词），191

T

table statistics（统计表），144
tags（标签），232
  cloud（云），232
tags（structured text）（标签（结构化文本）），357

task（任务）
ad hoc（随机），163
Chinese（中文），164
cross language（跨语言），164
filtering（过滤），164
high precision（高精度），164
interactive（交互），164
NLP（自然语言处理），164
spoken document retrieval（语音文档检索），164
very large corpus（超大规模语料库），164
taxonomies（分类体系），205
taxonomy（分类体系），231
building（构建），330
IR models（信息检索模型），59
term（项，索引项，查询项）
correlation（相关），68，96，100
correlation factor（相关因子），196
expansion（扩展），32
highlighting（加亮显示），31
incidence contingency table（出现列联表），82
neighborhoods（邻域），192
probabilistic reweighting（概率再赋权），183
probabilistic weight（概率权重），80
reweighting（再赋权），181
vector weight（向量权重），77
vector weighting（向量赋权），72
weighting（赋权），66
weights（权重），66，196
term correlation factor（项相关因子），100
term partitioning（项划分），405
term vector（项向量），100
term-document matrix（项文档矩阵），62，340
term-frequency（项频），参见 TF
term-term correlation matrix（项间相关性矩阵），67，96，190
termsets（项集），87
closed（闭），91
text（文本），203
entropy（熵），219
full text（全文），63
mining（挖掘），49
properties（属性），218
separators（分隔符），219
similarity（相似度），222
snippets（片段），30
structure（结构），256
text classification（文本分类），281
algorithms（算法），284
characterization（特性），282
clustering（聚类），286
cross-validation（交叉检验），329
decision trees（决策树），294
ensemble classifier（集成分类器），316
evaluation（评价），291
evaluation metrics（评价指标），325
F-measure（F 值），327
k-NN（*k* 近邻），299
naïve（朴素），290
naive Bayes（朴素贝叶斯），303
precision（准确率，精度），327
problem definition（问题定义），283
recall（召回率），327
standard collections（标准文档集），329
supervised algorithms（监督算法），291
SVM（支持向量机），306
unsupervised algorithms（无监督算法），286
text compression（文本压缩），233
arithmetic coding（算术编码），239
coding（编码），234，238
comparing techniques（比较技术），248
dense coding（密集编码），242
dictionary（词典），234
dictionary methods（词典方法），234，245
direct searching（直接搜索），241，244
Huffman coding（霍夫曼编码），238
in IR（信息检索），253
inverted index structure（倒排索引结构），340
modeling（建模），234
adaptive（自适应），235
high-order（高阶），236
semi-static（半静态），235
static（静态），235
word-based（基于词的），236
zero-order（零阶），236
phrase（短语），234
preprocessing（预处理），246
statistical（统计），234
statistical methods（统计方法），234
structured text（结构化文本），249
LZCS（LZCS 压缩方法），250
SCM（结构化上下文建模压缩方法），250
XMLPPM（XMLPPM 压缩方法），249
text languages（文本语言）
XForms（XForms 下一代互联网表单），251
text model（文本模型），218
Text REtrieval Conference（文本检索会议），159
text structure（文本结构）
fixed（固定），263
hierarchical（层次），264

hypertext（超文本），264
text structure，field（文本结构，域），263
textual annotations（文本注释），653
texture（纹理），593
contrast（对比度），596
energy（能量），595
entropy（熵），595
homogeneity（均匀性），596
TF（项频），68
TF-IDF（项频-反比文档频率）
normalized factors（归一化因子），93
properties（性质），74
variants（变体），73
weights（权重），68
thesaurus（同义词典），96，193，228
built-in（内置），103
concept（概念），229
keywords（关键词），229
similarity（相似度），195
statistical（统计），198
term relationships（项间关系），230
use（应用），230
Thompson's algorithm（Thompson 算法），382
thumbnails（缩略图），46
timbre（音色），603
TIME reference collection（TIME 参考文档集），167
TIPSTER（TIPSTER 文本计划），159
program（计划），160
tone height（音高），605
topic（主题），162
topic diffusion（主题扩散），471
topology（拓扑）
belief network（信念网），122
inference network（推理网），117
total frequency in collection（文档集总频率），67
total term frequency（项总频率），67
training set（训练集），291
TREC（文本检索会议），163，743，745
average precision histogram（平均精度直方图），166
benchmark（基准），163
document（文档），161
document collection（文档集），160
document level averages（文档等级平均精度），165
enterprise track（企业搜索任务），663
evaluation measures（评测指标），165
information request（信息需求），163
recall-precision averages（平均召回率-精度），165
summary table statistics（摘要统计表），165
Web site（网站），160
triangle inequality（三角不等式），222
trie（trie 树），361
trust bias（信赖偏好），187

U

unary code（一元码），355
uniterm system（Uniterm 系统），132
universal search（统一搜索），30，491
unsupervised learning（无监督学习），283
URI（统一资源标识符），206
URL（统一资源定位符），206
user（用户）
behavior（行为），186，269
goals（目标），50
judgement（判断），181
modeling（建模），677
need（需求），7
preferences through clicks（点击偏好），187
profile（轮廓），677
relevance feedback（相关反馈），180
task（任务），4，50
user-centered design（以用户为中心的设计），50

V

variants of TF-IDF（TF-IDF 变体），73
vector model（向量模型），60，77
advantages（优点），79
definition（定义），77
norms（范数），76
ranking（排序），78
vector term（向量项）
weighting（赋权），72
weights（权重），77
vertical crawler（垂直爬取），517
vertical search（垂直搜索），126
video retrieval（视频检索），606
abstracts（摘要），606
browsing（浏览），606
dynamic summaries（动态摘要），609
interactive summaries（交互式摘要），611
mosaics and salient stills（图像拼接与跳跃剧照），608
static summaries（静态摘要），607
storyboards（故事板），607
video segmentation（视频分割），620
edges（边缘），623
schemes（方案），622
virtual processing ratio（虚处理比），419
virtual processors（虚处理器），419

virtual reality（虚拟现实），208
visual browsing（视觉浏览），612
visualization（可视化）
  query terms in context（上下文中的查询项），43
  relationships among words anddocuments（词语和文档间的关系），47
  text mining（文本挖掘），49
visualization of search results（搜索结果可视化），40
VLC2 collection（VLC2 文档集），162
vocabulary（词汇表），62，221，237，340
  space（空间），343
vocabulary-set（词汇表集），87
VRML（虚拟现实建模语言），208，251

W

WAIS（广域信息服务），266
wavelet tree（小波树），371
WDSL（互联网服务描述语言），216
Web，8，447，451
  advertising（广告），504
  book Web site（本书网站），16
  bow-tie（蝴蝶结结构），452
  characteristics（特性），451
  characterization（刻画），518
  communities（社区），457
  components（分支），452
  core component（核心分支），452，453
  data mining（数据挖掘），506
  directories（目录），499
  duplicated pages（重复页面），479
  dynamic pages（动态页面），520
  history（历史），8
  hypertext（超文本），501
  impact on search（对搜索的影响），10
  largest strongly connected component（最大强连通分支），452
  link analysis（链接分析），456
  link graph compression（链接图压缩），478
  link graph structure（链接图结构），452
  metadata（元数据），459
  metasearch（元搜索），508
  practical issues（实际问题），12
  private（私有），520
  public（公共），520
  semantic（语义），450
  site analysis（网站分析），519
  spam（垃圾），参见 Web spam
  static pages（静态页面），520
Web crawler（Web 爬虫），参见 crawler
Web mining（Web 挖掘），506
  content（内容），506
  link structure（链接结构），507
  queries（查询），507
  usage（用途），507
Web retrieval（Web 检索），参见 Web search
Web search（Web 搜索），447
  actionable results（可执行的结果），496
  advanced search（高级搜索），481
  answers（答案），450
  architecture（架构），458
    centralized（集中式），458
    cluster based（基于集群的），459
    distributed（分布式），466
    multi-site（多站点），467
  assigning ids to documents（为文档分配标识符），477
  caching（缓存），462
    answers（答案），462
    inverted lists（倒排表），463
  content delivery（内容传送），512
  data quality（数据质量），449
  data volume（数据容量），449
  distributed data（分布式数据），449
  duplicated pages（重复页面），449
  dynamic data（动态数据），509
  effectiveness（有效性），486
  efficiency（效率），486
  heterogeneous data（异质数据），450
  metadata（元数据），478
  multimedia（多媒体），510
  multiple indexes（多级索引），464
  oneboxes（oneboxes 直接结果），490
  optimization（优化），477
  query facets（查询分面），495
  query languages（查询语言），482
  query log privacy（查询日志隐私性），512
  query recommendation（查询推荐），493
    content aware（内容感知），493
    content ignorant（内容无关），494
    hybrid（混合），494
    query-flow based（基于查询流的），494
  querying（查询），450
  ranking（排序），468
    authority（权威度），470
    content signals（内容信号），469
    HITS（超文本推导主题搜索），470
    hub（枢纽），471
    in-link count（入链计数），470
    learning a function（学习函数），474

learning to rank（排序学习），473
link-based（基于链接的），470
structure signals（结构信号），469
usage signals（用途信号），469
redundant data（冗余数据），449
result page（结果页面），488
semantic（语义），510
snippet（片段），489
spelling assistance（拼写帮助），493
spelling correction（拼写校对），486
structured data（结构化数据），449
suggestions（建议），484
deduplication（去重），486
diversity（多样性），487
filtering（过滤），486
freshness（新鲜度），487
personalization（个性化），487
teaching the user（培养用户）497
user interaction（用户交互），480
simplicity rule（简单性规则），480
volatile data（不稳定数据），449
web query languages（Web 查询语言），503
Web spam（Web 垃圾），476
click-based（基于点击的），477
content-based（基于内容的），477
link-based（基于链接的），477
Web-slang（Web 俚语），508
WebTREC（WebTREC 文档集），参见 WT10g
weighted information gain（加权信息增益），276
weights（权重）
IDF（反比文档频率），70
term-frequency（项频），68
TF-IDF（项频-反比文档频率），68
vector model（向量模型），77
Wikipedia（维基百科），491，495
wild characters（通配符），262
wisdom of crowds（群体智慧），507
word pattern（词汇模式），390
word similarity（单词相似度），260
word-based modeling（基于词汇的建模），参见 text compression，254
World Wide Web（万维网），参见 Web
World Wide Web consortium（万维网联盟），514
WT10g（WT10g 文档集），748
WT10g collection（WT10g 文档集），162，744
WT2g collection（WT2g 文档集），162

X

XForms（XForms 下一代互联网表单），251
XHTML（可扩展超文本标记语言），213
XML（可扩展标记语言），214
XML retrieval（XML 检索），545
content-and-structure topics（内容和结构主题），567
content-only topics（内容主题），567
document collections（文档集），566
dynamic structure（动态结构），547
evaluation（评价），566
measures（测度），571
precision（精度），572
recall（召回率），572
exhaustivity（穷尽性），570
explicit structure（显式结构），546
implicit structure（隐式结构），546
indexing（索引），553
INEX（INEX XML 检索评测），566
multiple hierarchies（多层次结构），548
non-overlapping lists（非覆盖列表），549
proximal nodes（相邻结点），550
query languages（查询语言），573
clause-based queries（子句查询），577
content constraints（内容约束），574
content-and-structure（内容和结构），575
content-only（内容），575
FLWOR（FLWOR 表达式），580
NEXI（NEXI 查询语言），578
path-based queries（路径查询），576
pattern matching constraints（模式匹配约束），574
structural constraints（结构约束），575
tag-based queries（标签查询），576
XPath（XML 路径语言），577
XQuery（XML 查询语言），579
XQuery Full-Text（XML 全文查询语言），581
ranking（排序），551，554
aggregation（聚合策略），560
contextualization（上下文化策略），558
element scoring（元素评分策略），555
merging（合并策略），562
overlaps（重叠），565
propagation（传播策略），558
structural constraints（结构化约束），563
relevance（相关性），569
single hierarchy（单一层次结构），548
specificity（特异性），570
static structure（固定结构），547
structuring power（结构化能力），546
tasks（任务），568

XMLPPM（XMLPPM 压缩方法），249，254
XPath（XML 路径语言），577
XQuery（XML 查询语言），579
XQuery Full-Text（XML 全文查询语言），581

Y

Yahoo（雅虎），481，482，484，486，491，492，504，506
  answers（问答），480，491，495
  assist（助手），485，486
  context match（上下文匹配），504
  directory（目录），452，499，501
  image search（图像搜索），488
  search（搜索），480，483，484，490，491，494-496
  search shortcuts（搜索快捷方式），491
Yule process（Yule 过程），457

Z

Z39.2（Z39.2 标准），689
Zipf's law（齐夫法则），71，220，237，455
Ziv-Lempel compression（Ziv-Lempel 压缩方法），245，254
zooming（缩放），42